Modeling Human Behavior at a Large Scale

by

Adam Sadilek

Submitted in Partial Fulfillment

of the

Requirements for the Degree

Doctor of Philosophy

Supervised by

Henry A. Kautz

Department of Computer Science
Arts, Science & Engineering
Edmund A. Hajim School of Engineering & Applied Sciences

University of Rochester
Rochester, New York

2012

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2012		**00-00-2012 to 00-00-2012**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Modeling Human Behavior at a Large Scale	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of Rochester,Rochester,NY,14611**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**197**	

To my family, for giving me a great starting point in life,
along with a balanced training set.

Curriculum Vitae

Adam Sadilek grew up in Prague, and received his Bachelor's degree from the Czech Technical University in 2008. He spent the 2006/2007 academic year as an exchange student at Union College in New York, and returned to the United States in 2008 to pursue a doctorate in Computer Science at the University of Rochester. Adam earned a Master of Science degree from the University of Rochester in 2010. Advised by Henry Kautz, Adam focuses on scalable artificial intelligence models that help us understand and predict emergent global phenomena, such as human behavior and disease epidemics, from day-to-day interactions of individuals. Along with his studies, Adam has done research and development work as an intern at eBay Research Labs, Google, and Microsoft Research, where he focused on optimization and novel applications of machine learning in the context of big data and location-based reasoning. Adam served on program committees of major conferences, including AAAI and IJCAI, and won the Best Paper Award at WSDM 2012, McKinsey&Company Scholarship, Accenture and Open Society Fellowship, Davenport Research Fellowship, and a number of full scholarships based on academic merit.

Acknowledgments

To my advisor Henry Kautz, and to my committee as a whole, Jeffrey Bigham, Lenhart Schubert, Vincent Silenzio, and Daniel Štefankovič: thank you for your guidance.

This research was partly funded by ARO grant W911NF-08-1-0242, ONR grant N00014-11-10417, OSD grant W81XWH-08-C0740, NSF SES-1110625 and NSF IIS-1012017. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of any of these organizations.

Abstract

Until recently, complex phenomena—such as human behavior and disease epidemics—have been modeled primarily at an aggregate level. Detailed studies have been limited to small domains encompassing only a few subjects, as scaling the methods involved poses considerable challenges in terms of cost, human effort required, computational bottlenecks, and data sources available. With the surge of online social media and sensor networks, the abundance of interesting and publicly accessible data is beginning to increase. However, we also need the ability to reason about it efficiently. The underlying theme of this thesis is the unification and data mining of diverse, noisy, and incomplete sensory data over large numbers of individuals. We show that the mined patterns can be leveraged in predictive models of human behavior and other phenomena at a large scale. We find that raw sensory data linked with the content of users' online communication, the explicit as well as the implicit online social interactions, and interpersonal relationships are rich information sources upon which strong machine learning models can be built. Example domains where such models apply include understanding human activities, predicting people's location and social ties from their online behavior, and predicting the emergence of global epidemics from day-to-day interpersonal interactions.

Table of Contents

List of Tables

List of Figures

1 Foreword

Chapter 4 is based on our work published in Sadilek and Kautz (2010a,b, 2012a,b). Chapter 5 is based on collaboration with John Krumm while at Microsoft Research (Sadilek and Krumm, 2012). Chapter 6 includes our work previously published in Sadilek *et al.* (2012a). Finally, Chapter 7 builds on results reported in Sadilek *et al.* (2012b,c).

2 Introduction

With the boom of information technology we experience today comes an explosion in the amount and richness of *data* recorded. The necessary ingredients for this phenomenon have been fully realized only recently: ubiquitous Internet connectivity, virtually limitless data storage, and powerful mobile devices. Crucially, all the ingredients are inexpensive, widely available, and the technology behind them reached a level of maturity where the general population—not just a handful of hackers—uses the devices in everyday life. As a result, Internet-enabled phones and other mobile computers are used by nearly everyone, even in less developed parts of the world. A typical phone has a large array of sensors that can record location, orientation, acceleration, light intensity, Bluetooth and Wi-Fi neighborhoods, temperature, and of course audio. Images and video can often be captured as well.

At the same time, the *social* aspects of computing are gaining prominence. According to a recent Nielsen study,[1] the average Facebook user spent nearly 7 hours and 45 minutes on the site per month—more than on any other single site on the Internet. To put these statistics in context, the average amount of time a person spends online per month is 30 hours.

Since more than 30% of smart phone users access online social networks from their phone,[2] some of the fine-grained sensory data the mobile devices record is now linked to the rich structured data in people's online profiles. This includes the text of their

[1] http://blog.nielsen.com/nielsenwire/online_mobile/august-2011-top-us-web-brands

[2] http://www.comscore.com/

messages, tags attached to photos and status updates, and the structure of their social network. For instance, we will see that a large fraction of online communication is geo-tagged with precise GPS coordinates and interlinked with information about related people, their location, their friends' location and content, and so on.

Since most of us carry a phone in our pockets virtually all the time, and since a large fraction of the population participates in online social media, it becomes possible to quantify—at a planetary scale—phenomena that have been elusive until now. As we will see, predicting the spread of disease is a specific instance of phenomena in this class. For example, 1 in 30 New York City residents appears in the Twitter dataset we collected in May 2010. This sampling ratio increases over time as more people join the social network and with more extensive data collection periods. This means that we have real-time access to detailed information about a significant fraction of the population. We can additionally view these individuals as "noisy sensors" of their surroundings, thereby enabling inference even about phenomena not directly recorded online or in government statistics.

A combination of machine learning and crowd sourcing is now poised to effectively answer questions that, at present, require years of laborious and expensive data collection. Moreover, we are no longer limited by the static and coarse-grained nature of the traditional statistics. However, new challenges are introduced as well. This thesis focuses on methods that overcome these challenges and enable us to capture important phenomena with a significantly increased level of detail, timeliness, and predictive power.

The underlying theme of our work is the *unification* and *data mining* of diverse, noisy, and incomplete sensory data over large numbers of individuals. We show that the mined patterns can be leveraged in *predictive* models of human behavior at a large scale. We find that the raw sensory data linked with the content of users' online communication, the explicit as well as the implicit online social interactions, and relationships are extremely rich information sources. Furthermore, this data is now available to machines in massive volumes and at ever-increasing real-time streaming rate.

The first portion of this thesis concentrates on modeling human behavior based on

location data in the form of raw GPS coordinates collected at regular intervals (*e.g.*, every second) by our subjects carrying GPS loggers. Perhaps surprisingly, rich models—both descriptive and predictive—of people's activities can be learned from this data source alone. Specifically, we show that we can reliably recognize fine-grained multi-agent activities in the game of capture the flag, even though the signal to noise ratio is very low. We then turn to location prediction and show that our "eigenday" model describes the typical day of a person in a meaningful way and enables accurate location prediction, even far into the future.

In the second half of the thesis, we transition from pure location-based domains to richer—but more noisy, irregular, and incomplete—datasets derived from online social media, including Twitter and FourSquare. The social and textual components of these data sources allow us to model phenomena, such as the disease epidemics, that would be out of reach for pure location-based systems. As we mentioned, accurate location is often embedded in this online data as well, as a large fraction of users access online services from their phones that often automatically attach GPS tags to user messages and photos.

We will see that people's social ties as well as their location are readily predictable from their online interactions. We then show that by unifying GPS traces revealed in people's online messages, along with text analysis of those messages, enables us to predict the progress of a contagion from person to person at a population scale and without any active user participation. Additionally, by leveraging the explicit social structure recorded in the data, we begin to quantify answers to important questions in public health. This includes the relationship between the number of sick people you encounter and the probability of you contracting the disease in the future, and the impact of the health of your social milieu on your own heath.

3 Background

This chapter reviews the context of the probabilistic models developed and applied in this thesis. We discuss both relational and propositional types of models, directed and undirected graphical models, as well as generative and discriminative. Additionally, we review decision trees, support vector machine classifiers, and introduce Twitter—a popular online social network.

3.1 Markov Logic

The cores of our activity recognition models described below are implemented in Markov logic (ML), a statistical-relational language. In this section, we provide a brief overview of ML, which extends finite first-order logic (FOL) to a probabilistic setting. For a more detailed (and excellent) treatment of FOL, ML, and inductive logic programming see the work of Shoenfield (1967); Domingos *et al.* (2008), and De Raedt and Kersting (2008), respectively.

Given the inherent uncertainty involved in reasoning about real-world activities as observed through noisy sensor readings, we looked for a methodology that would provide an elegant combination of probabilistic reasoning with the expressive, relatively natural, and compact but unfortunately strictly true or false formulas of first-order logic. And that is exactly what Markov logic provides and thus allows us to elegantly model complex finite relational non-i.i.d. domains. A Markov logic network (MLN) consists of a set of constants $\mathcal{C}$ and of a set of pairs $\langle \mathcal{F}_i, w_i \rangle$ such that each FOL formula $\mathcal{F}_i$ has a weight

$w_i \in \mathbb{R}$ associated with it. Optionally, each weight can be further scaled by a real-valued function of a subset of the variables that appear in the corresponding formula. Markov logic networks that contain such functions are called *hybrid* MLNs (Wang and Domingos, 2008).

A MLN can be viewed as a template for a Markov network (MN) as follows: the MN contains one node for each possible ground atom of MLN. The value of the node is 0 if the corresponding atom is *false* and 1 otherwise. Two nodes are connected by an edge if the corresponding atoms appear in the same formula. Thus, the MN has a distinct clique corresponding to each grounding of each formula. By $\mathcal{F}_i^{g_j}$ we denote the j-th grounding of formula $\mathcal{F}_i$. The MN has a feature value $f_{i,j}$ for each $\mathcal{F}_i^{g_j}$ such that

$$f_{i,j} = \begin{cases} 1 & \text{if } \mathcal{F}_i^{g_j} \text{ is } \textit{true} \\ 0 & \text{otherwise} \end{cases}$$

Each weight w_i intuitively represents the relative "importance" of satisfying (or violating, if the weight is negative) the corresponding formula $\mathcal{F}_i$. More formally, the weight scales the difference in log-probability between a world that satisfies n groundings of the corresponding formula and one that results in m true groundings of the formula, all else being equal (*cf.* Equation 3.1). Thus the problem of satisfiability is relaxed in MLNs. We no longer search for a satisfying truth assignment as in traditional FOL. Instead, we are looking for a truth assignment that maximizes the sum of the weights of all satisfied formulas.

The weights can be either specified by the knowledge base engineer or, as in our approach, learned from training data. That is, we provide the learning algorithm with labeled capture instances and pairs of raw and corresponding denoised trajectories along with labeled instances of game events and it finds an optimal set of weights that maximize the likelihood of the training data. Weight learning can be done in either generative or discriminative fashion. Generative training maximizes the joint probability of observed (evidence) as well as hidden (query) predicates, whereas discriminative learning directly maximizes the conditional likelihood of the hidden predicates given the observed predicates. Since prior work demonstrated that Markov network models learned

discriminatively consistently outperform their generatively trained counterparts (Singla and Domingos, 2005), we focus on discriminative learning in our activity recognition domain.

Once the knowledge base with weights has been specified, we can ask questions about the state of hidden atoms given the state of the observed atoms. Let X be a vector of random variables (one random variable for each possible ground atom in the MN) and let $\boldsymbol{\chi}$ be the set of all possible instantiations of X. Then, each $x \in \boldsymbol{\chi}$ represents a possible world. If $(\forall x \in \boldsymbol{\chi})[\Pr(X = x) > 0]$ holds, the probability distribution over these worlds is defined by

$$\Pr(X = x) = \frac{1}{Z} \exp\left(\sum_i w_i n_i\left(x_{\{i\}}\right)\right) \tag{3.1}$$

where $n_i(x_{\{i\}})$ is the number of true groundings of i-th formula with w_i as its weight in a world x and

$$Z = \sum_{x \in \boldsymbol{\chi}} \exp\left(\sum_i w_i n_i\left(x_{\{i\}}\right)\right). \tag{3.2}$$

Equation 3.1 can be viewed as assigning a "score" to each possible world and dividing each score by the sum of all scores over all possible worlds (the constant Z) in order to normalize.

Maximum *a posteriori* (MAP) inference in Markov logic given the state of the observed atoms reduces to finding a truth assignment for the hidden atoms such that the weighed sum of satisfied clauses is maximal. Even though this problem is NP-complete, we achieve reasonable run times by applying Cutting Plane MAP Inference (CPI) (Riedel, 2008). CPI can be thought of as a meta solver that incrementally grounds a Markov logic network, at each step creating a Markov network that is subsequently solved by any applicable method—such as MaxWalkSAT or via a reduction to an integer linear program. CPI refines the current solution by searching for additional groundings that could contribute to the objective function.

Up to this point, we have focused on *first-order* Markov logic. In first-order ML, each variable ranges over objects present the domain (*e.g.*, apples, players, or cars).

On the other hand, in finite *second-order* Markov logic, we variabilize not only objects but also predicates (relations) themselves (Kok and Domingos, 2007b). Our CTF model contains a predicate variable for each *type* of activity. For example, we have one variable *captureType* whose domain is {capturing, failedCapturing} and analogously for freeing events. When grounding the second-order ML, we ground all predicate variables as well as object variables. There has also been preliminary work on generalizing ML to be well-defined over infinite domains, which would indeed give it the full power of FOL (Singla and Domingos, 2007).

Implementations of Markov logic include Alchemy[1] and theBeast[2]. Our experiments used a modified version of theBeast.

3.2 Probabilistic Models of Sequential Data

Much of our work concentrates on spatio-temporal sequences of data. In this section, we review the context of probabilistic models that are amiable to this type of data, and discuss, in more detail, several specific instances of this family of models we heavily use in our work.

Two broad classes of probabilistic models appear in machine learning literature: *generative* and *discriminative*. Generative models aim at compactly representing a full joint probability distribution $\Pr(x, y)$ where x and y denote vectors of *observed* (*e.g.*, data coming directly from the sensors, such as current GPS reading) and *hidden* (*e.g.*, the actual true location of the player) random variables, respectively. The well-known hidden Markov model (HMM) is an example generative model.

On the other hand, discriminative models deal with conditional probability distributions in the form of $\Pr(y \mid x)$. Since our goal is to detect and recognize "hidden" events from GPS data and not to, for instance, create synthetic GPS traces, we do not need to model the statistical dependencies among the sensor data x. The state of x is always known and we only need to infer the state of the hidden variables y. This "specializa-

[1] `http://alchemy.cs.washington.edu/`

[2] `http://code.google.com/p/theBeast/`

tion" renders the inference problem more tractable while requiring less training data, since we have to consider and model fewer parameters and dependencies. A conditional random field (CRF) is a type of discriminative model.

Further, we distinguish between *directed* and *undirected* probabilistic models. This nomenclature is based on the observation that the models can be conceptually represented as a graph where each node of the graph corresponds to a random variable and for each pair of nodes, there is an edge between them if the corresponding random variables are dependent in some way. If these edges are directed, we say that the model is a directed model and otherwise, we say it is a undirected model. A dynamic Bayesian network (DBN) is an example of directed model, whereas a Markov random field (MRF) is an undirected model.

A variety of specific probabilistic models has been applied to activity recognition. There are trade-offs to consider when choosing a particular model for the problem at hand. For instance in the case of DBNs, cyclicity of the the model is problematic, and further, the parameters of the model are conditional probability tables, which are quite unnatural for people to specify or review. By contrast, in MRFs, we can encode the parameters more intuitively by defining a set of feature functions f_i (one for each (type of) clique i in the graph), such that each function maps from the state of the random variables in its domain to a non-negative real value. A weighted sum of these functions then defines the probability distribution induced by a particular MRF as follows.

$$P(X = x, Y = y) = \frac{1}{Z} \exp\left(\sum_i w_i f_i\left(x_{\{i\}}, y_{\{i\}}\right)\right), \tag{3.3}$$

where Z denotes a partition function (a normalizing constant) given by

$$Z = \sum_{x \in \boldsymbol{\chi}} \exp\left(\sum_i w_i f_i\left(x_{\{i\}}, y_{\{i\}}\right)\right). \tag{3.4}$$

Thus, MRFs allow us to define quite complex parameters of our models in a natural way and, as opposed to DBNs, they facilitate modeling long-range (irregular) dependencies.

3.2.1 Dynamic Bayesian Networks

A Bayesian network (BN) is a directed probabilistic graphical model (Jordan, 1998). Nodes in the graph represent random variables and edges represent conditional dependencies (*cf.* Figure 4.4). For a BN with n nodes, the joint probability distribution is given by

$$\Pr(X_1, \ldots, X_n) = \prod_{i=1}^{n} \Pr\big(X_i | \mathrm{Pa}(X_i)\big), \tag{3.5}$$

where $\mathrm{Pa}(X_i)$ denotes the parents of node X_i. In a typical setting, a subset of the random variables is *observed* (we know their actual values), while the others are *hidden* and their values need to be inferred.

A dynamic Bayesian network (DBN) is a BN that models sequential data. A DBN is composed of *slices*—in our case each slice represents a one second time interval. In order to specify a DBN, we either write down or learn intra- and inter-slice conditional probability distributions (CPDs). The intra-slice CPDs typically constitute the observation model while the inter-slice CPDs model transitions between hidden states. For an extensive treatment of DBNs, see the work of Murphy (2002).

There are a number of parameter learning and inference techniques for DBNs. To match the Markov logic-based framework, in the experiments with the DBN model presented below, we focus on a supervised learning scenario, where the hidden labels are known at training time and therefore a maximum likelihood estimate can be calculated directly.

We find a set of parameters (discrete probability distributions) θ that maximize the log-likelihood of the training data. This is achieved by optimizing the following objective function.

$$\theta^{\star} = \operatorname*{argmax}_{\theta} \ \log\big(\Pr\big(x_{1:t}, y_{1:t} | \theta\big)\big), \tag{3.6}$$

where $x_{1:t}$ and $y_{1:t}$ represent the sequence of observed and hidden values, respectively, between times 1 and t, and $\theta^{\star}$ is the set of optimal model parameters. In our implemen-

tation, we represent probabilities and likelihoods with their log-counterparts to avoid arithmetic underflow.

At testing time, we are interested in the most likely explanation of the observed data. That is, we want to calculate the most likely assignment of states to all the hidden nodes (*i.e.*, Viterbi decoding of the DBN) given by

$$y^{\star}_{1:t} = \underset{y_{1:t}}{\operatorname{argmax}} \ \log\big(\Pr(y_{1:t}|x_{1:t})\big), \tag{3.7}$$

where $\Pr(y_{1:t}|x_{1:t})$ is the conditional probability of a sequence of hidden states $y_{1:t}$ given a concrete sequence of observations $x_{1:t}$ between times 1 and t. We calculate the Viterbi decoding efficiently using dynamic programming (Jordan, 1998).

We apply DBNs in our work because they naturally model time series data (time flows in one direction), we can highly optimize both learning and inference. Since the hidden nodes in our models are discrete, we perform both parameter learning and exact inference efficiently by customized versions of the Baum-Welch algorithm and Viterbi decoding, respectively. For a detailed treatment of these methods see (Jordan, 1998). We exaplain how we apply DBNs to our Twitter domain in Section 6.6.2.

3.2.2 Conditional Random Fields

A conditional random field (CRF) is a discriminative undirected graphical model (Lafferty, 2001). CRFs have been successfully applied in a wide range of domains from language understanding (Sha and Pereira, 2003; Roark *et al.*, 2004) to robotics (Ramos *et al.*, 2007). They systematically outperform alternative approaches, such as hidden Markov models, in domains where it is unrealistic to assume that observations are independent given the hidden state. Instead of applying Bayes' theorem to estimate the distribution over hidden states y given observations x, CRFs explicitly encode the *conditional* distribution $\Pr(y|x)$.

CRFs factorize the conditional distribution into a product of *clique potentials* as follows

$$\Pr(y|x) = \frac{1}{Z(x)} \prod_{c \in C} \phi_c(x_c, y_c) \tag{3.8}$$

$$Z(x) = \sum_{y} \prod_{c \in C} \phi_c(x_c, y_c), \tag{3.9}$$

where C is the set of cliques within the graph that defines the structure of the CRF, and x_c and y_c are the observed and hidden nodes, respectively, in a particular clique c.

Since evaluation of the normalizing constant $Z(x)$ is exponential in the size of y, exact inference is feasible only in small domains or for specific classes of CRF structures, such as trees.

In practice, the clique potentials ϕ are further decomposed into a log-linear weighted combination of *feature functions* f:

$$\phi_c(x_c, y_c) = \exp\Big\{w_c^T f_c(x_c, y_c)\Big\}. \tag{3.10}$$

We can now express the conditional probability in terms of feature functions

$$\Pr(y|x) = \frac{1}{Z(x)} \exp\left\{\sum_{c \in C} w_c^T f_c(x_c, y_c)\right\}. \tag{3.11}$$

Learning and inference in conditional random fields applies analogous techniques discussed in the previous section on DBNs.

3.3 Support Vector Machines

Support vector machine (SVM) is an established model of data in machine learning (Cortes and Vapnik, 1995). In this work, we learn SVMs for linear binary classification that accurately distinguish between tweets indicating the author is afflicted by an infectious ailment (we call such tweets "sick"), and all other tweets (called "other" or "normal").

Linear binary SVMs are trained by finding a hyperplane defined by a normal vector w with the maximal margin separating it from the positive and negative datapoints. Finding such a hyperplane is inherently a quadratic optimization problem given by the following objective function

$$\min_{w} \frac{\lambda}{2}||w||^2 + \mathcal{L}(w, D), \tag{3.12}$$

where λ is a regularization parameter controlling model complexity, and $\mathcal{L}(w, D)$ is the hinge-loss over all training data D given by

$$\mathcal{L}(w, D) = \sum_i \max\left(0, 1 - y_i w^T x_i\right). \tag{3.13}$$

The optimization problem in Equation 3.12 can be solved efficiently and in a parallel fashion using stochastic gradient descend methods (Shalev-Shwartz *et al.*, 2007).

Class imbalance, where the number of examples in one class is dramatically larger than in the other class, complicates virtually all machine learning. For SVMs, prior work has shown that transforming the optimization problem from the space of individual datapoints $\langle x_i, y_i \rangle$ in matrix D to one over *pairs* of examples $\left\langle x_i^+ - x_j^-, 1 \right\rangle$ yields significantly more robust results (Joachims, 2005). (x_i^+ denotes feature vectors from the positive class ($y_i = +1$), whereas x_j^- denotes negatively labeled data points ($y_j = -1$).)

This method is often referred to as ROC Area SVM because it directly maximizes the area under the ROC curve of the model. By bundling examples into pairs, we effectively find a w that minimizes the number n_s of incorrectly ranked (swapped) pairs in the training data given by

$$n_s = \left|\left\{(i, j) : (y_i > y_j) \wedge (w^T x_i < w^T x_j)\right\}\right|. \tag{3.14}$$

At testing time, a feature vector x_t is classified simply by a dot product multiplication with the weight vector: $\tilde{y} = \text{sign}(w^T x_t)$.

3.4 Decision Trees

Decision trees are models of data encoded as rules induced from examples (Breiman and others, 1984). Intuitively, in the Twitter domain, a decision tree represents a series of questions that need to be asked and answered in order to estimate the probability of friendship between any two people, based on their attributes. During decision tree learning, features are evaluated in terms of information gain with respect to the labels

and the best candidates are subsequently selected for each inner node of the tree. Our implementation uses *regression* decision trees, where each leaf contains the probability of a friendship. As described below, we also employ decision trees for feature selection, since they intrinsically rank features by their information content.

3.5 Twitter

Our social network experiments are based on data obtained from Twitter, a popular micro-blogging service where people post message updates at most 140 characters long. The forced brevity encourages frequent mobile updates. Relationships between users on Twitter are not necessarily symmetric. One can follow (subscribe to receive messages from) a user without being followed back. When users do reciprocate following, we say they are *friends* on Twitter. There is anecdotal evidence that Twitter friendships have a substantial overlap with offline friendships (Gruzd *et al.*, 2011). Twitter launched in 2006 and has been experiencing an explosive growth since then. As of June 2011, over 300 million accounts are registered on Twitter. FourSquare and Gowalla are the most prominent location-based social networks that allow people to share their location, and to "check-in" at various venues near them. A subset of these check-ins appears in our Twitter dataset, as some users link their Twitter accounts with one or more other services.

4 Relational Activity Recognition

Figure 4.1: A snapshot of a game of capture the flag that shows most of the game area. Players are represented by pins with letters. In our version of CTF, the two "flags" are stationary and are shown as white circles near the top and the bottom of the figure. The horizontal road in the middle of the image is the territory boundary. The data is shown prior to any denoising or corrections for map errors.

4.1 Overview

Recent research has shown that surprisingly rich models of human activity can be learned from GPS (positional) data. However, most effort to date has concentrated on modeling single individuals or statistical properties of groups of people. Moreover, prior work focused solely on modeling actual successful executions (and not failed or attempted executions) of the activities of interest. We, in contrast, take on the task of understanding human interactions, attempted interactions, and intentions from noisy sensor data in a *fully relational multi-agent setting*.

We use a real-world game of capture the flag to illustrate our approach in a well-defined domain that involves many distinct cooperative and competitive joint activities. We model the domain using Markov logic, a statistical-relational language, and learn a theory that jointly denoises the data and infers occurrences of high-level activities, such as a player capturing an enemy. Our unified model combines constraints imposed by the geometry of the game area, the motion model of the players, and by the rules and dynamics of the game in a probabilistically and logically sound fashion.

We show that while it may be impossible to directly detect a multi-agent activity due to sensor noise or malfunction, the occurrence of the activity can still be inferred by considering both its impact on the future behaviors of the people involved as well as the events that could have preceded it. Further, we show that given a model of successfully performed multi-agent activities, along with a set of examples of failed attempts at the same activities, our system automatically learns an augmented model that is capable of recognizing success and failure, as well as goals of people's actions with high accuracy.

We compare our approach with other alternatives and show that our unified model, which takes into account not only relationships among individual players, but also relationships among activities over the entire length of a game, although more computationally costly, is significantly more accurate. Finally, we demonstrate that explicitly modeling unsuccessful attempts boosts performance on other important recognition tasks.

4.2 Motivation

Our society is founded on the interplay of human relationships and interactions. Since every person is tightly embedded in our social structure, the vast majority of human behavior can be fully understood only in the context of the actions of others. Thus, not surprisingly, more and more evidence shows that when we want to model behavior of a person, the single best predictor is often the behavior of people in her social network. For instance, behavioral patterns of people taking taxis, rating movies, choosing a cell phone provider, or sharing music are best explained and predicted by the habits of related people, rather than by all the "single person" attributes such as age, race, or education (Bell *et al.*, 2007; Pentland, 2008).

In contrast to these observations, most research effort on activity recognition to date has concentrated on modeling single individuals (Bui, 2003; Liao *et al.*, 2004, 2005), or statistical properties of aggregate groups of individuals (Abowd *et al.*, 1997; Horvitz *et al.*, 2005), or combinations of both (Eagle and Pentland, 2006). Notable exceptions to this "isolated individuals" approach includes the work of Kamar and Horvitz (2009) and Gupta *et al.* (2009), where simple relationships among people are just starting to be explicitly considered and leveraged. For instance, Eagle and Pentland (2006) elegantly model the location of individuals from multi-modal sensory data, but their approach is oblivious to the explicit effects of one's friends, relatives, *etc.* on one's behavior. The isolated individuals approximations are often made for the sake of tractability and representational convenience. While considering individuals independently of each other is sufficient for some constrained tasks, in many interesting domains it discards a wealth of important information or results in an inefficient and unnatural data representation. On the other hand, decomposing a domain into a set of entities (representing for instance people, objects in their environment, or activities) that are linked by various relationships (*e.g.*, is-a, has-a, is-involved-in) is a natural and clear way of representing data.

To address the shortcomings of nonrelational behavior modeling, we introduce the capture the flag domain (described below), and argue for a statistical-relational approach

to learning models of multi-agent behavior from raw GPS data. The CTF dataset is on one hand quite complex and recorded by real-world sensors, but at the same time it is well-defined (as per the rules of the game), thereby allowing for an unambiguous evaluation of the results.

Being able to recognize people's activities and reason about their behavior is a necessary precondition for having intelligent and helpful machines that are aware of "what is going on" in the human-machine as well as human-human relationships. There are many exciting practical applications of activity recognition that have the potential to fundamentally change people's lives. For example, cognitive assistants that help people and teams be more productive, or provide support to (groups of) disabled individuals, or efficiently summarize a long complex event to a busy person without leaving out essential information. Other important applications include intelligent navigation, security (physical as well as digital), human-computer interaction, and crowdsourcing. All these applications and a myriad of others build on top of multi-agent activity recognition and therefore require it as a necessary stepping stone. Furthermore, as a consequence of the anthropocentrism of our technology, modeling human behavior plays—perhaps surprisingly—a significant role even in applications that do not directly involve people (*e.g.*, unmanned space probes).

Furthermore, reasoning about human *intentions* is an essential element of activity recognition, since if we can recognize what a person (or a group of people) *wants* to do, we can proactively try to help them (or—in adversarial situations—hinder them). Intent is notoriously problematic to quantify (Baldwin and Baird, 2001, *e.g.*,), but we show that in the capture the flag domain, the notion is naturally captured in the process of learning the structure of failed activities. We all know perhaps too well that a successful action is often preceded—and unfortunately sometimes also followed—by multiple failed attempts. Therefore, reasoning about attempts typically entails high practical utility, but not just for their relatively high frequency. Consider, for example, a task of real-time analysis of a security video system. There, detecting that a person or a group of people (again, relations) *intend* to steal something is much more important and useful than recognizing that a theft has taken (or even is taking) place, because then it is

certainly too late to entirely *prevent* the incident, and it may also be too late or harder to merely stop it. We believe that recognition of attempts in people's activities is a severely underrepresented topic in artificial intelligence that needs to be explored more since it opens a new realm of interesting possibilities.

Before we delve into the details of our approach in Sections 4.5 and 4.6, we briefly introduce the CTF dataset (Section 4.3), highlight the main contributions of our work (Section 4.4). Background material has been reviewed in Chapter 3. We discuss related work, conclude, and outline future work in Sections 4.7, 4.8 and 4.9 respectively.

4.3 Capture The Flag Domain

Imagine two teams—seven players each—playing capture the flag (CTF) on a university campus, where each player carries a consumer-grade global positioning system (GPS) that logs its location (plus noise) every second (see Figure 4.1). The primary goal is to enter the opponent's flag area. Players can be captured only while on enemy territory by being tagged by the enemy. Upon being captured, they must remain in place until freed (tagged by a teammate) or the game ends. The games involve many competitive and cooperative activities, but here we focus on (both successful and attempted) capturing and freeing. Visualization of the games is available from the first author's website.

We collected four games of CTF on a portion of the University of Rochester campus (about 23 acres) with Columbus V-900 GPS loggers (one per player) with 1 GB memory card each that were set to a sampling rate of 1 Hz. The durations of the games ranged approximately from 4 to 15 minutes.

Our work is not primarily motivated by the problem of annotating strategy games, although there are obvious applications of our results to sports and combat situations. We are, more generally, exploring relational learning and inference methods for recognizing *multi-agent activities* from location data. We accept the fact that the GPS data at our disposal is inherently unreliable and ambiguous for any one individual. We therefore focus on methods that *jointly and simultaneously* localize and recognize the high-level activities of groups of individuals.

Although the CTF domain doesn't capture all the intricacies of life, it contains many complex, interesting, and yet well-defined (multi-agent) activities. Moreover, it is based on extensive real-world GPS data (total of 40,000+ data points). Thus most of the problems that we are addressing here clearly have direct analogs in everyday-life situations that ubiquitous computing needs to address—imagine people going about their daily lives in a city instead of CTF players, and their own smart phones instead of GPS loggers.

One of the main challenges we have to overcome if we are to successfully model CTF is the severe noise present in the data. Accuracy of the GPS data varies from 1 to more than 10 meters. In open areas, readings are typically off by 3 meters, but the discrepancy is much higher in locations with tall buildings (which are present within the game area) or other obstructions. Compare the scale of the error with the granularity of the activities we concern ourselves with: both capturing and freeing involves players that are within reaching distance (less than 1 meter) apart. Therefore, the signal to noise ratio in this domain is daunting.

The error has a systematic component as well as a significant stochastic component. Errors between devices are poorly correlated, because subtle differences between players, such as the angle at which the device sits in the player's pocket, can dramatically affect accuracy. Moreover, since we consider multi-agent scenarios, the errors in individual players' readings can add up, thereby creating a large discrepancy between the reality and the recorded dataset. Because players can move freely through open areas, we cannot reduce the data error by assuming that the players move along road or walkways, as is done in much work on GPS-based activity recognition (Liao *et al.*, 2004, *e.g.*,). Finally, traditional techniques for denoising GPS data, such as Kalman filtering, are of little help, due to the low data rate (1 sample per second) relative to the small amount of time required for a player to completely change her speed or direction.

If we are to reliably recognize events that happen in these games in the presence of such severe noise, we need to consider not only each player, but also the relationships among them and their actions over extended periods of time (possibly the whole length of the game). Consider a concrete task of inferring the individual and joint activities

Figure 4.2: Three snapshots of a game situation where both successful and failed capturing occur. This example also illustrates the need for an approach that exploits both the relational and the far reaching temporal structure of our domain. (See text for explanation.)

and intentions of the CTF players from their GPS traces. For example, suppose the GPS data shows player A running toward a stationary teammate B, then moving away. What occurred? Possibly player A has just "freed" player B, but GPS error has hidden the fact that player A actually *reached* B. Another possibility is that player A had the *intention* of freeing player B, but was scared off by an opponent at the last second. Yet another possibility is that no freeing occurred nor was even intended, because player B had not been previously captured.

Understanding a game thus consists of inferring a complex set of interactions among the various players as well as the players' intentions. The conclusions drawn about what occurs at one point in time affect and are affected by inferences about past and future events. In the example just given, recognizing that player B is moving in the future reinforces the conclusion that player A is freeing player B, while failing to recognize a past event of player B being captured decreases confidence in that conclusion. The game of CTF also illustrates that understanding a situation is as much or more about recognizing attempts and intentions as about recognizing successfully executed actions. For example, in course of a 15 minute game, only a handful of capture or freeing events occur. However, there are dozens of cases where one player unsuccessfully tries to capture an opponent or to free a teammate. A description of a game that was restricted to what actually occurred would be only a pale reflection of the original.

As a concrete example, consider a real game situation illustrated in Figure 4.2. There we see three snapshots of a game projected over a map of the campus before any modification of the GPS data. The game time is shown on each snapshot. Players D, F, and G are allies and are currently on their home territory near their flag, whereas players L and M are their enemies. In the first snapshot, players L and M head for the opponent's flag but then—in the second frame—they are intercepted by G. At this point it is unclear what is happening because of the substantial error in the GPS data—the three players appear to be very close to each other, but in actuality they could have been 20 or more meters apart. However, once we see the third snapshot (note that tens of seconds have passed) we realize that player G actually captured only player M and didn't capture L since G is evidently still chasing L. The fact that player M remains stationary coupled with the fact that neither D nor F attempt to capture him suggests that M has indeed been captured. We show that it is possible to infer occurrences of capturing events even for complex situations like these whereas limited approaches largely fail. However, we need to be able to recognize not just individual events, we also need to discover new activities, identify their respective goals, and distinguish between events based on whether their outcomes are favorable or negative. For instance, in the second frame, player G tries to capture both L and M. Although he succeeded in the former case, he failed in the latter.

Many different kinds of cooperative and competitive multi-agent activities occur in the games. The lowest-level joint activities are based on location and movement, and include "approaching" and "being at the same location." Note, that noise in the GPS data often makes it difficult or impossible to directly detect these simple activities. At the next level come competitive multi-agent activities including capturing and attacking; cooperative activities include freeing; and there are activities, such as chasing and guarding, that may belong to either category or to both categories. There are also more abstract tactical activities, such as making a sacrifice, and overall strategies, such as playing defensively. In this chapter, we concentrate on activities at the first two levels.

4.4 Our Contributions

The main contributions of this chapter are as follows. We first present a novel method that simultaneously denoises positional data and learns a model of multi-agent activities that occur there. We subsequently evaluate the model on the CTF dataset and show that it achieves high accuracy in recognizing complex game events.

However, creating a model by manually writing down new rules or editing existing axioms is laborious and prone to introduction of errors or unnecessarily complex theories. Thus, we would like to automate this process by *learning* (or *inducing*) new axioms from training data. For people, it is much easier to provide or validate concrete examples than to directly modify a model. This leads us to our second contribution: We show how to automatically augment a preexisting model of (joint) activities so that it is capable of not only recognizing successful actions, but also identifies *failed attempts* at the same types of activities. This line of work also demonstrates that explicitly modeling attempted interactions in a unified way improves overall model performance.

As our third contribution, we demonstrate that the *difference* (discussed below) between the newly learned definitions of a failed activity and the original definition of the corresponding successful activity directly corresponds to the *goal* of the given activity. For instance, as per the rules of the capture the flag game, a captured player cannot move until freed. When our system induces the definition of failed capture, the new theory does not contain such a constraint on the movement of the almost-captured player, thereby allowing him to move freely.

4.5 Methodology

In this section, we describe the three major components of our approach. In short, we first manually construct a model of captures and freeings in CTF and optimize its parameters in a supervised learning framework (Section 4.5.1). This constitutes our "seed" theory that is used for denoising raw location data and recognition of successful multi-agent activities. We then show, in Section 4.5.2, how to automatically extend the

seed theory by inducing the structure and learning the importance of failed captures and freeings as well as the relationships to their successful counterparts. Finally, in Section 4.5.3, we use the augmented theory to recognize this richer set of multi-agent activities—both successful and failed attempts—and extract the goals of the activities.

Specifically, we investigate the following four research questions:

Q1. Can we reliably recognize complex multi-agent activities in the CTF dataset even in the presence of severe noise?

Q2. Can models of attempted activities be automatically learned by leveraging existing models of successfully performed actions?

Q3. Does modeling both success and failure allow us to infer the respective goals of the activities?

Q4. Does modeling failed attempts of activities improve the performance on recognizing the activities themselves?

We now elaborate on each of the three components of our system in turn, and subsequently discuss, in light of the experimental results and lessons learned, our answers to the above research questions.

4.5.1 Recognition of Successful Activities

In this section, we present our unified framework for intelligent relational denoising of the raw GPS data while simultaneously labeling instances of a player being captured by an enemy or freed by an ally. Both the denoising and the labeling are cast as a learning and inference problem in Markov logic. By denoising, we mean modifying the raw GPS trajectories of the players such that the final trajectories satisfy constraints imposed by the geometry of the game area, the motion model of the players, as well as by the rules and the dynamics of the game. In this chapter, we refer to this trajectory modification as "snapping" since we tile the game area with 3 by 3 meter cells and snap each raw GPS reading to an appropriate cell. By creating cells only in unobstructed space, we ensure the final trajectory is consistent with the map of the area.

We begin by modeling the domain via a Markov logic theory, where we write the logical formulas that express the structure of the model by hand, and learn an optimal set of weights on the formulas from training data in a supervised discriminative fashion (details on the experimental set-up are in Section 4.6). In the following two subsections, we will show how to augment this seed Markov logic theory to recognize a richer set of events and extract the goals of players' multi-agent activities.

In order to perform data denoising and recognition of successful capturing and freeing, we model the game as weighted formulas in Markov logic. Some of the formulas are "hard," in the sense that we are only interested in solutions that satisfy all of them. Hard formulas capture basic physical constraints (*e.g.*, a player is only at one location at a time) and inviolable rules of the game (*e.g.*, a captured player must stand still until freed or the game ends).[1] The rest of the formulas are "soft," meaning there is a finite weight associated with each one. Some of the soft constraints correspond to a traditional low-level data filter, expressing preferences for smooth trajectories that are close to the raw GPS readings. Other soft constraints capture high-level constraints concerning when individual and multi-agent activities *are likely* to occur. For example, a soft constraint states that if a player encounters an enemy on the enemy's territory, the player is likely to be captured. The exact weights on the soft constraints are learned from labeled data, as described below.

We distinguish two types of atoms in our models: *observed* (*e.g.*, GPS(P_1, 4, 43.13°, −77.71°) and *hidden* (*e.g.*, freeing(P_1, P_8, 6)). The observed predicates in the CTF domain are: GPS, enemies, adjacent, onHomeTer, and onEnemyTer;[2] whereas capturing, freeing, isCaptured, isFree, samePlace, and snap are hidden. Additionally, the set of hidden predicates is expanded by the structure learning algorithm described below (see Table 4.1 for predicate semantics). In the training phase, our learning algorithm has access to the known truth assignment to *all* atoms. In the testing phase, it can still

[1] Cheating did not occur in our CTF games, but in principle could be accommodated by making the rules highly-weighted soft constraints rather than hard constraints.

[2] While the noise in the GPS data introduces some ambiguity to the last two observed predicates, we can still reliably generate them since the road that marks the boundary between territories constitutes a neutral zone.

Hard Rules:

H1. Each raw GPS reading is snapped to exactly one cell.

H2. (a) When player a frees player b, then both involved players must be snapped to a common cell at that time.

(b) A player can only be freed by a free ally.

(c) A player can be freed only when he or she is currently captured.

(d) Immediately after a freeing event, the freed player transitions to a free state.

(e) A player can only be freed while on enemy territory.

H3. (a) When player a captures player b, then both involved players must be snapped to a common cell at that time.

(b) A player can only be captured by a free enemy.

(c) A player can be captured only if he or she is currently free.

(d) Immediately after a capture event, the captured player transitions to a captured state.

(e) A player can be captured only when standing on enemy territory.

H4. All players are free at the beginning of the game.

H5. At any given time, a player is either captured or free but not both.

H6. A player transitions from a captured state to a free state only via a freeing event.

H7. A player transitions from a free state to a captured state only via a capture event.

H8. If a player is captured then he or she must remain in the same location.

Soft Rules:

S1. Minimize the distance between the raw GPS reading and the snapped-to cell.

S2. Minimize projection variance, *i.e.*, two consecutive "snappings" should be generally correlated.

S3. Maximize smoothness (both in terms of space and time) of the final player trajectories.

S4. If players a and b are enemies, a is on enemy territory and b is not, b is not captured already, and they are close to each other, then a *probably* captures b.

S5. If players a and b are allies, both are on enemy territory, b is currently captured and a is not, and they are close to each other, then a *probably* frees b.

S6. Capture events are generally rare, *i.e.*, there are typically only a few captures within a game.

S7. Freeing events are also generally rare.

Figure 4.3: Descriptions of the hard and soft rules for capture the flag.

Predicate	**Type**	**Meaning**
capturing(a, b, t)	hidden	Player a is capturing b at time t.
enemies(a, b)	observed	Players a and b are enemies.
adjacent(c_1, c_2)	observed	Cells c_1 and c_2 are mutually adjacent, or $c_1 = c_2$.
failedCapturing(a, b, t)	hidden	Player a is unsuccessfully capturing b at time t.
failedFreeing(a, b, t)	hidden	Player a is unsuccessfully freeing b at time t.
freeing(a, b, t)	hidden	Player a is freeing b at time t.
isCaptured(a, t)	hidden	Player a is in captured state at time t.
isFailedCaptured(a, t)	hidden	At time t, player a is in a state that follows an unsuccessful attempt at capturing a. a in this state has the same capabilities as when free.
isFailedFree(a, t)	hidden	At time t, player a is in a state that follows an unsuccessful attempt at freeing a. a in this state has the same capabilities as when captured.
isFree(a, t)	hidden	Player a is in free state at time t (isFree$(a, t) \equiv \neg$ isCaptured(a, t)).
onEnemyTer(a, t)	observed	Player a in on enemy territory at time t.
onHomeTer(a, t)	observed	Player a in on home territory at time t.
samePlace(a, b, t)	hidden	Players a and b are either snapped to a common cell or to two adjacent cells at time t.
snap(a, c, t)	hidden	Player a is snapped to cell c at time t.

Table 4.1: Summary of the logical predicates our models use. Predicate names containing the word "failed" are introduced by the Markov logic theory augmentation method described in Section 4.5.2.

access the state of the observed atoms, but it has to infer the assignment to the hidden atoms.

Figure 4.3 gives an English description of our hard and soft rules for the low-level movement and player interactions within capture the flag. Corresponding formulas in the language of ML are shown in Figures 4.5 and 4.6.

We compare our unified approach with four alternative models. The first two models (**baseline** and **baseline with states**) are purely deterministic and they separate the denoising of the GPS data and the labeling of game events. We implemented both of them in Perl. They do not involve any training phase. The third alternative model is a **dynamic Bayesian network** shown in Figure 4.4. Finally, we have two models cast in Markov logic: the **two-step ML model** and the **unified ML model** itself. The unified model handles the denoising and labeling in a joint fashion, whereas the two-step approach first performs snapping given the geometric constraints and subsequently labels instances of capturing and freeing. The latter three models are evaluated using four-fold cross-validation where in order to test on a given game, we first train a model on the other three games.

All of our models can access the following observed data: raw GPS position of each player at any time and indication whether they are on enemy or home territory, location of each 3 by 3 meter cell, cell adjacency, and list of pairs of players that are enemies. We tested all five models on the same observed data. The following describes each model in more detail.

- **Baseline Model (B)**

  This model has two separate stages. First we snap each reading to the nearest cell and afterward we label the instances of player a capturing player b. The labeling rule is simple: we loop over the whole discretized (via snapping) data set and output capturing(a, b, t) every time we encounter a pair of players a and b such that they were snapped (in the first step) to either the same cell or to two mutually adjacent cells at time t, they are enemies, and a is on its home territory while b is not. Freeing recognition is not considered in this simple model since

we need to have a notion of persisting player states (captured or free) in order to model freeing in a meaningful way.

- **Baseline Model with States (B+S)**

  This second model builds on top of the previous one by introducing a notion that players have states. If player a captures player b at time t, b enters a captured state (in logic, isCaptured$(b, t+1)$). Then b remains in captured state until he moves (is snapped to a different cell at a later time) or the game ends. As per rules of CTF, a player who is in captured state cannot be captured again.

  Thus, this model works just like the previous one except whenever it is about to label a capturing event, it checks the states of the involved players and outputs capturing(a, b, t) only if both a and b are *not* in captured state.

  Freeing recognition is implemented in an analogous way to capturing recognition. Namely, every time a captured player b is about to transition to a free state, we check if b has a free teammate a nearby (again, within the adjacent cells). If that is the case, we output freeing(a, b, t).

- **Dynamic Bayesian Network Model (DBN)**

  The dynamic Bayesian network model can be viewed as a probabilistic generalization of the above baseline model with states. The structure of the DBN model for one player is shown in Figure 4.4. In each time slice, we have one hidden node and four observed nodes, all of which represent binary random variables. We want to infer the most likely state S for each player at any given time t over the course of a game. The state is either free or captured and is hidden at testing time. There are four observed random variables per time step that model player's motion (M), presence or absence of at least one enemy (EN) and ally (AN) player nearby, and finally player's location on either home or enemy territory (ET). Each player is modeled by a separate DBN. Therefore, there are fourteen instantiated DBNs for each game, but within any one game, all the DBNs share the same set of parameters.

  Note that the DBN model does not perform any GPS trajectory denoising itself.

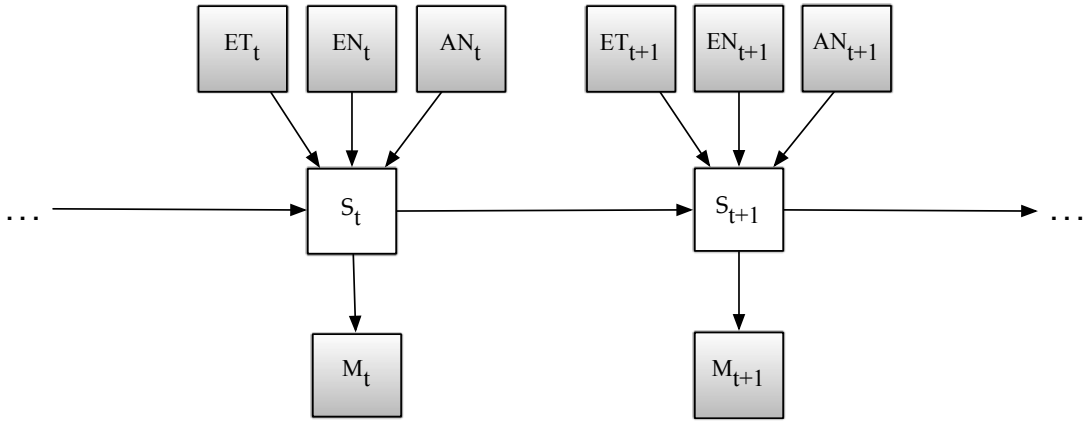


Figure 4.4: Two consecutive time slices of our dynamic Bayesian network for modeling the state of an individual player P from observations. Shaded nodes represent observed random variables, unfilled denote hidden variables. All random variables are binary. ($ET_t = 1$ when P is on enemy territory at time t, $EN_t = 1$ when there is an enemy nearby at time t, $AN_t = 1$ when there is an ally nearby at time t, and finally $M_t = 1$ if P has moved between time $t-1$ and t. The value of hidden state S_t is 1 if P is captured at time t and 0 when P is free.)

To make a fair comparison with the Markov logic models, we use the denoising component of the Markov logic theory using only constraints H1 and S1–S3 (in Figure 4.3). This produces a denoised discretization of the data that is subsequently fed into the DBN model. The random variables within the DBN that capture the notion of player "movement" and players being "nearby" one another is defined on the occupancy grid of the game area, just like in the two deterministic baseline models. Namely, a player is said to be moving between time t and $t+1$ when he or she is snapped to two different nonadjacent cells at those times. Similarly, two players are nearby if they are snapped either to the same cell or to two adjacent cells.

- **Two-Step ML Model (2SML)**

  In the two-step approach, we have two separate theories in Markov logic. The first theory is used to perform a preliminary snapping of each of the player tra-

jectories individually using constraints H1 and S1–S3 (in Figure 4.3). This theory is identical to the one used in the discretization step in the DBN model above.

The second theory then takes this preliminary denoising as a list of observed atoms in the form preliminarySnap(a, c, t) (meaning player a is snapped to cell c at time t) and uses the remaining constraints to label instances of capturing and freeing, while considering cell adjacency in the same manner as the previous three models. The two-step model constitutes a decomposition of the unified model (see below) and overall contains virtually the same formulas, except 2SML operates with an observed preliminarySnap predicate, whereas the unified model contains a hidden snap predicate instead. Thus we omit elaborating on it further here.

- **Unified ML Model (UML)**

  In the unified approach, we express all the hard constraints H1–H8 and soft constraints S1–S7 (Figure 4.3) in Markov logic as a single theory that jointly denoises the data and labels game events. Selected interesting formulas are shown in Figure 4.6—their labels correspond to the listing in Figure 4.3. Note that formulas S1–S3 contain real-valued functions d_1, d_2, and d_3 respectively. d_1 returns the distance between agent a and cell c at time t. Similarly, d_2 returns the dissimilarity of the two consecutive "snapping vectors"[3] given agent a's position at time t and $t+1$ and the location of the centers of two cells c_1 and c_2. Finally, since people prefer to move in straight lines, function d_3 quantifies the lack of smoothness of any three consecutive segments of the trajectory. Since w_p, w_s, and w_t are all assigned negative values during training, formulas S1–S3 effectively softly enforce the corresponding geometric constraints.

  The presence of functions d_1 through d_3 renders formulas S1–S3 hybrid formulas. This means that at inference time, the instantiated logical part of each formula evaluates to either 1 (true) or 0 (false), which is in turn multiplied by the product of the corresponding function value and the formula weight.

[3]The initial point of each snapping (projection) vector is a raw GPS reading and the terminal point is the center of the cell we snap that reading to.

We will see how we train, test, and evaluate these four models, and how they perform on the multi-agent activity recognition task in Section 4.6. Next, we turn to our supervised learning method for augmenting the unified ML model in order to recognize both successful and failed attempts at multi-agent activities.

4.5.2 Learning Models of Failed Attempts

In the work described above, we manually designed the structure of a Markov logic network that models the capture the flag domain and allows us to jointly denoise the raw GPS data and recognize instances of actual capturing and freeing. Now we show how to *automatically*—in a supervised learning setting—extend this theory to encompass and correctly label not only successful actions, but also failed attempts at those interactions. That is, given the raw GPS data that represent the CTF games, we want our new model to label instances where player a captures (or frees) player b as *successful captures* (*successful frees*) and instances where player a almost captures (or frees) player b as *failed captures* (*failed frees*). For example, by "failed capturing" we mean an instance of players' interactions where—up to a point—it appeared that a is capturing b, but when we carefully consider the events that (potentially) preceded it as well as the impacts of the supposed capture on the future unfolding of the game, we conclude that it is a false alarm and no capture actually occurred. In other words, the conditions for a capture were right, but later on, there was a pivotal moment that foiled the capturing agent's attempt.

For both activities (capturing and freeing), our model jointly finds an optimal separation between success and failure. Note that since we cast our model in second-order Markov logic, we do not learn, *e.g.*, an *isolated* rule that separates successful freeing from a failed attempt at freeing. Rather—since capturing and freeing events (both actual and failed) are related and thus labeling an activity as, say, "successful capturing" has far-reaching impact on our past, present, and future labeling—we learn the separations in a joint and unified way. Namely, both the structure (logical form) and importance (weight) of each formula in our theory is considered with all its consequences and influence on other axioms in the theory. Our system thus finds an optimal balance between

Hard formulas:

(H1) $$\forall a, t\ \exists c : \text{snap}(a, c, t)$$
$$\forall a, c, c', t : (\text{snap}(a, c, t) \wedge c \neq c') \Rightarrow \neg\text{snap}(a, c', t)$$

(H2) $$\forall a_1, a_2, t : \text{freeing}(a_1, a_2, t) \Rightarrow \big(\text{samePlace}(a_1, a_2, t) \wedge \text{isFree}(a_1, t) \wedge \neg\text{enemies}(a_1, a_2) \wedge \text{isCaptured}(a_2, t) \wedge \text{isFree}(a_2, t+1) \wedge \text{onEnemyTer}(a_1, t) \wedge \text{onEnemyTer}(a_2, t)\big)$$

(H3) $$\forall a_1, a_2, t : \text{capturing}(a_1, a_2, t) \Rightarrow \big(\text{samePlace}(a_1, a_2, t) \wedge \text{isFree}(a_1, t) \wedge \text{enemies}(a_1, a_2) \wedge \text{isFree}(a_2, t) \wedge \text{isCaptured}(a_2, t+1) \wedge \text{onHomeTer}(a_1, t) \wedge \text{onEnemyTer}(a_2, t)\big)$$

$$\forall a_1, a_2, t : \text{samePlace}(a_1, a_2, t) \Rightarrow \big(\exists c_1, c_2 : \text{snap}(a_1, c_1, t) \wedge \text{snap}(a_2, c_2, t) \wedge \text{adjacent}(c_1, c_2)\big)$$

(H4) $$\forall a, t : (t = 0) \Rightarrow \text{isFree}(a, t)$$

(H5) $$\forall a, t : \text{isCaptured}(a, t) \oplus \text{isFree}(a, t)$$

(H6) $$\forall a, t : (\text{isFree}(a, t) \wedge \text{isCaptured}(a, t+1)) \Rightarrow (\exists_{=1} a_1 : \text{capturing}(a_1, a, t))$$

(H7) $$\forall a, t : (\text{isCaptured}(a, t) \wedge \text{isFree}(a, t+1)) \Rightarrow (\exists_{=1} a_1 : \text{freeing}(a_1, a, t))$$

(H8) $$\forall a, t, c : (\text{isCaptured}(a, t) \wedge \text{isCaptured}(a, t+1) \wedge \text{snap}(a, c, t)) \Rightarrow \text{snap}(a, c, t+1)$$

Figure 4.5: Our hard formulas in Markov logic. See corresponding rules in Figure 4.3 for an English description and Table 4.1 for explanation of the predicates. In our implementation, the actual rules are written in the syntax used by theBeast, a Markov logic toolkit. ($\exists_{=1}$ denotes unique existential quantification, $\oplus$ designates exclusive or.)

Soft formulas:

(S1) $$\forall a, c, t : \big[\text{snap}(a, c, t)\big] \cdot d_1(a, c, t) \cdot w_p$$

(S2) $$\forall a, c_1, c_2, t : \big[\text{snap}(a, c_1, t) \wedge \text{snap}(a, c_2, t+1)\big] \cdot d_2(a, c_1, c_2, t) \cdot w_s$$

(S3) $$\forall a, c_1, c_2, c_3, t : \big[\text{snap}(a, c_1, t) \wedge \text{snap}(a, c_2, t+1) \wedge \text{snap}(a, c_3, t+2)\big] \cdot d_3(a, c_1, c_2, c_3, t) \cdot w_t$$

(S4) $$\forall a_1, a_2, t : [(\text{enemies}(a_1, a_2) \wedge \text{onHomeTer}(a_1, t) \wedge \text{onEnemyTer}(a_2, t) \wedge \text{isFree}(a_2, t) \wedge \text{samePlace}(a_1, a_2, t)) \Rightarrow \text{capturing}(a_1, a_2, t)] \cdot w_c$$

(S5) $$\forall a_1, a_2, t : [(\neg\text{enemies}(a_1, a_2) \wedge \text{onEnemyTer}(a_1, t) \wedge \text{onEnemyTer}(a_2, t) \wedge \text{samePlace}(a_1, a_2, t) \wedge \text{isFree}(a_1, t) \wedge \text{isCaptured}(a_2, t)) \Rightarrow \text{freeing}(a_1, a_2, t)] \cdot w_f$$

(S6) $$\forall a, c, t : \big[\text{capturing}(a, c, t)\big] \cdot w_{cb}$$

(S7) $$\forall a, c, t : [\text{freeing}(a, c, t)] \cdot w_{fb}$$

Figure 4.6: Soft formulas in Markov logic. See corresponding rules in Figure 4.3 for an English description. Each soft formula is written as a traditional quantified finite first-order logic formula (*e.g.*, $\forall a, c, t : \big[\text{snap}(a, c, t)\big]$), followed by an optional function (*e.g.*, $d_1(a, c, t)$), followed by the weight of the formula (*e.g.*, w_p). This syntax denotes that at inference time, the instantiated logical part of each formula evaluates to either 1 (true) or 0 (false), which is then effectively multiplied by the product of corresponding function value and formula weight.

success and failure in capturing and freeing activities with respect to the training data.

The Theory Augmentation Algorithm

In what follows, we will describe our Markov logic theory augmentation algorithm (Algorithm 1). For clarity, we will explain how it works in concrete context of the ML models of capture the flag we discussed in previous sections. However, the underlying assumption that successful actions are in many ways similar to their failed counterparts, and that minor—but crucial—deviations cause the failure to occur, often hold beyond capture the flag. Therefore, the same algorithm is applicable to other domains with different activities, as long as they are modeled in Markov logic.

Algorithm 1 : Extend a ML theory to model successful as well as failed activities.

Input: A: set of activities

$\mathcal{M}_S$: ML theory that models successful instances of activities in A

S: set of examples of successful activities

F: set of examples of failed activities

Output: $\mathcal{M}_{S+F}$: augmented ML model with learned weights that models both successful and attempted activities in A

$\mathcal{I}$: intended goals of the activities

1: $\mathcal{M}_S^2 \Leftarrow \text{liftToSecondOrderML}(\mathcal{M}_S, A)$

2: $\mathcal{M}'_S \Leftarrow \text{instantiate}(\mathcal{M}_S^2, A)$

3: $\mathcal{I} \Leftarrow \text{findIncompatibleFormulas}(F, \mathcal{M}'_S)$

4: $\mathcal{M}_{S+F} \Leftarrow \mathcal{M}'_S \setminus \mathcal{I}$

5: $\mathcal{M}_{S+F} \Leftarrow \text{learnWeights}(S, F, \mathcal{M}_{S+F})$

6: $\mathcal{M}_{S+F} \Leftarrow \text{removeZeroWeightedFormulas}(\mathcal{M}_{S+F})$

7: **return** $\mathcal{M}_{S+F}, \mathcal{I}$

At a high-level, the augmentation algorithm belongs to the family of structure learning methods. Starting with a seed model of successful actions, it searches for new formulas that can be added to the seed theory in order to jointly model both successfully and unsuccessfully carried out actions. The declarative language bias—essentially rules

for exploring the hypothesis space of candidate structures—is defined implicitly by the notion that for any given activity, the structure of unsuccessful attempts is similar to the successful attempts. Therefore, the augmentation algoritm goes through an "inflation" stage, where formulas in the seed theory are generalized, followed by a refinement stage, where superfluous and incompatible formulas in the inflated model are pruned away. The refinement step also optimizes the weights within the newly induced theory. We will now discuss this process in more detail.

The input of our theory augmentation algorithm consists of an initial first-order ML theory $\mathcal{M}_S$ that models successful capturing and freeing (such as the unified ML model defined in Section 4.5.1 that contains formulas shown in Figures 4.5 and 4.6), a set of activities of interest A, and a set of examples of successful (S) as well as failed (F) captures and frees. $\mathcal{M}_S$ does not need to have weights for its soft formulas specified. In case they are missing, we will learn them from scratch in the final steps of the augmentation algorithm. If the weights are specified, the final weight learning step for $\mathcal{M}_{S+F}$ can leverage them to estimate the initial weight values. A can be specified as a set of predicate names, *e.g.*, {capturing, freeing}. Each example in sets S and F describes a game segment and constitutes a truth assignment to the appropriate literals instantiated from $\mathcal{M}_S$. Table 4.2 shows two toy examples of sets S and F for three time steps. Since the goal is to learn a model of failed (and successful) attempts in a supervised way, the example game segment in F contain activities labeled with predicates failedCapturing() and failedFreeing().

If $\mathcal{M}_S$ contains hybrid formulas (such our formulas S1–S3 in Figure 4.6), the appropriate function definitions are provided as part of S and F as well. Each definition consists of implicit mapping from input arguments to function values. For instance, function d_1 in formula S1 quantifies the L2 distance between the agent a and cell c at time t in the projected Mercator space: $d_1(a, c, t) = \sqrt{(a.gpsX_t - c.gpsX)^2 + (a.gpsY_t - c.gpsY)^2}$.

Our system goes through the following process in order to induce a new theory $\mathcal{M}_{S+F}$ that augments $\mathcal{M}_S$ with a definition of failed attempts for each activity already defined in $\mathcal{M}_S$.

First we lift $\mathcal{M}_S$ to second-order Markov logic by variabilizing all predicates that

Set S: Successful Capture	**Set F: Failed Capture**
enemies(P_1, P_2)	enemies(P_4, P_5)
enemies(P_2, P_1)	enemies(P_5, P_4)
	onEnemyTer($P_5, 1$)
onEnemyTer($P_2, 2$)	onEnemyTer($P_5, 2$)
onEnemyTer($P_2, 3$)	onEnemyTer($P_5, 3$)
capturing($P_1, P_2, 2$)	failedCapturing($P_4, P_5, 2$)
isFree($P_1, 1$)	isFree($P_4, 1$)
	isFailedCaptured($P_4, 1$)
isFree($P_1, 2$)	isFree($P_4, 2$)
	isFailedCaptured($P_4, 2$)
isFree($P_1, 3$)	isFree($P_4, 3$)
	isFailedCaptured($P_4, 3$)
isFree($P_2, 1$)	isFree($P_5, 1$)
	isFailedCaptured($P_5, 1$)
isFree($P_2, 2$)	isFree($P_5, 2$)
	isFailedCaptured($P_5, 2$)
isCaptured($P_2, 3$)	isFree($P_5, 3$)
	isFailedCaptured($P_5, 3$)
snap($P_1, C5, 1$)	snap($P_4, C17, 1$)
snap($P_1, C10, 2$)	snap($P_4, C34, 2$)
snap($P_1, C10, 3$)	snap($P_4, C0, 3$)
snap($P_2, C9, 1$)	snap($P_5, C6, 1$)
snap($P_2, C10, 2$)	snap($P_5, C34, 2$)
snap($P_2, C10, 3$)	snap($P_5, C7, 3$)
samePlace($P_1, P_2, 2$)	samePlace($P_4, P_5, 2$)
samePlace($P_2, P_1, 2$)	samePlace($P_5, P_4, 2$)
samePlace($P_1, P_2, 3$)	
samePlace($P_2, P_1, 3$)	

Table 4.2: Two examples of a logical representation of successful (S) as well as failed (F) capture events that are input to Algorithm 1. The closed-world assumption is applied, therefore all atoms not listed are assumed to be false. For clarity, we omit listing the adjacent() predicate.

correspond to the activities of interest (step 1 of Algorithm 1). This yields a lifted theory $\mathcal{M}_S^2$. More concretely, in order to apply this technique in our domain, we introduce new predicate variables *captureType* (whose domain is {capturing, failedCapturing}), *freeType* (over {freeing, failedFreeing}), and *stateType* (over {isCaptured, isFailedCaptured, isFree, isFailedFree}). For instance, variabilizing a first-order ML formula $\text{freeing}(a, b, t) \Rightarrow \neg\text{enemies}(a, b)$ yields a second-order ML formula $\textit{freeType}(a, b, t) \Rightarrow \neg\text{enemies}(a, b)$ (note that *freeType* is now a variable). Instantiating back to first-order yields two formulas: $\text{freeing}(a, b, t) \Rightarrow \neg\text{enemies}(a, b)$ and $\text{failedFreeing}(a, b, t) \Rightarrow \neg\text{enemies}(a, b)$.

As far as agents' behavior is concerned, in the CTF domain, isCaptured is equivalent to isFailedFree, and isFree is equivalent to isFailedCaptured. As we will soon see, the theory augmentation process learns these equivalence classes and other relationships between states from training examples by expanding and subsequently refining formula H5 in Figure 4.5. While we could work with only the isCaptured predicate and its negation to represent agents' states, we feel that having explicit failure states makes our discussion clearer. Furthermore, future work will need to address hierarchies of activities, including their failures. In that context, a representation of explicit failure states may not only be convenient, but may be necessary.

Next, we instantiate all predicate variables in $\mathcal{M}_S^2$ to produce a new first-order ML theory $\mathcal{M}_S'$ that contains the original theory $\mathcal{M}_S$ in its entirety plus new formulas that correspond to failed captures and frees (step 2). Since events that are, *e.g.*, near-captures appear similar to actual successful captures, our hypothesis is that we do not need to drastically modify the original "successful" formulas in order to model the failed activities as well. In practice, the above process of lifting and instantiating indeed results in a good seed theory. While we could emulate the lifting and grounding steps with a scheme of copying formulas and renaming predicates in the duplicates appropriately, we cast our approach in principled second-order Markov logic, which ties our work more closely to previous research and results in a more extensible framework. Specifically, second-order Markov logic has been successfully used in deep transfer learning (Davis and Domingos, 2009) and predicate invention (Kok and Domingos, 2007a). Therefore, an interesting

direction of future work is to combine our theory augmentation and refinement with transfer and inductive learning—operating in second-order ML—to jointly induce models of failed attempts of different activities in different domains, while starting with a single model of only successful activities in the source domain.

Typical structure learning and inductive logic programming techniques start with an initial (perhaps empty) theory and iteratively grow and refine it in order to find a form that fits the training data well. In order to avoid searching the generally huge space of hypotheses, a declarative bias is either specified by hand or mined from the data. The declarative bias then restricts the set of possible refinements of the formulas that the search algorithm can apply. Common restrictions include limiting formula length, and adding a new predicate to a formula only when it shares at least one variable with some predicate already present in the formula. On the other hand, in our approach, we first generate our seed theory by instantiating all the activity-related predicate variables. To put it into context of structure learning, we expand the input model in order to generate a large seed theory, and then apply bottom-up (data-driven) learning to prune the seed theory, whereby the training data guides our search for formulas to remove as well as for an optimal set of weights on the remaining formulas. We conjecture that any failed attempt at an activity always violates at least one constraint that holds for successful executions of the activity. The experiments below support this conjecture.

The pruning is done in steps 3 and 4 of Algorithm 1. The function findIncompatibleFormulas(F, $\mathcal{M}'_S$) returns a set of hard formulas in $\mathcal{M}'_S$ that are *incompatible* with the set of examples of failed interactions F. We say that a formula c is compatible with respect to a set of examples F if F logically entails c ($F \models c$). Conversely, if F does not entail c, we say that c is incompatible w.r.t. F. We explain how to find incompatible formulas in the next section.

In step 4 of Algorithm 1, we simply remove all incompatible formulas ($\mathcal{I}$) from the theory. At this point, we have our $\mathcal{M}_{S+F}$ model, where hard formulas are guaranteed logically consistent with the examples of failed activities (because we removed the incompatible hard formulas), as well as with the successful activities (because they were logically consistent to start with). However, the soft formulas in $\mathcal{M}_{S+F}$ are missing

properly updated weights (in Markov logic, the weight of each hard formula is simply set to $+\infty$). Therefore, we run Markov logic weight learning using theBeast package (step 5).

Recall that theBeast implements the cutting plane meta solving scheme for inference in Markov logic, where the ground ML network is reduced to an integer linear program that is subsequently solved by the LpSolve ILP solver. We chose this approach as opposed to, *e.g.*, MaxWalkSAT that may find a solution that is merely locally optimal, since the resulting run times are still relatively short (under an hour even for training and testing even the most complex model). Weights are learned discriminatively, where we directly model the posterior conditional probability of the hidden predicates given the observed predicates. We set theBeast to optimize the weights of the soft formulas via supervised on-line learning using margin infused relaxed algorithm (MIRA) for weight updates while the loss function is computed from the number of false positives and false negatives over the hidden atoms. Note that if any of the soft formulas are truly irrelevant with respect to the training examples, they are not picked out by the findIncompatibleFormulas() function, but their weights are set to zero (or very close to zero) in the weight learning step (line 5 in Algorithm 1). These zero-weighted formulas are subsequently removed in the following step. Note that the weight learning process does not need to experience a "cold" start, as an initial setting of weights can be inherited from the input theory $\mathcal{M}_S$.

Finally, we return the learned theory $\mathcal{M}_{S+F}$, whose formulas are optimally weighted with respect to *all* training examples. In the Experiments and Results section below, we will use $\mathcal{M}_{S+F}$ to recognize both successful and failed activities. Algorithm 1 also returns the incompatible hard formulas $\mathcal{I}$. We will see how $\mathcal{I}$ is used to extract the intended goal of the activities in the Section 4.5.3, but first, let us discuss step 3 of Algorithm 1 in more detail.

Consistency Check: Finding Incompatible Formulas

Now we turn to our method for finding incompatible formulas (summarized in Algorithm 2). Since our method leverages satisfiability testing to determine consistency

between candidate theories and possible worlds (examples),[4] Algorithm 2 can be viewed as an instance of learning from interpretations—a learning setting in the inductive logic programming literature (De Raedt, 2008).

Algorithm 2 (findIncompatibleFormulas). Find formulas in a ML theory that are logically inconsistent with examples of execution of failed activities.

Input: F: a set of examples of failed activities

$\mathcal{T}$: unrefined ML theory of successful and failed activities

Output: smallest set of formulas that appear in $\mathcal{T}$ and are unsatisfiable in the worlds in F

1: $O \Leftarrow \text{extractObjects}(F)$
2: $\mathcal{T}_{\text{hard}} \Leftarrow \mathcal{T} \setminus \mathcal{T}_{\text{soft}}$
3: **integer** $n \Leftarrow 0$
4: **boolean** result $\Leftarrow$ **false**
5: **while** result == **false do**
6: $\quad \mathcal{T}^c \Leftarrow \mathcal{T}_{\text{hard}}$
7: $\quad$ remove a new n-tuple of formulas from $\mathcal{T}^c$
8: $\quad$ **if** for the current n, all n-tuples have been tested **then**
9: $\quad\quad n \Leftarrow n + 1$
10: $\quad$ **end if**
11: $\quad$ result $\Leftarrow$ testSAT(F, $\mathcal{T}^c$, O)
12: **end while**
13: **return** $\mathcal{T}_{\text{hard}} \setminus \mathcal{T}^c$

As input, we take a set of examples of failed activities F and a seed theory $\mathcal{T}$ (*e.g.*, produced in step 2 of Algorithm 1). The output is the smallest set of hard formulas that appear in $\mathcal{T}$ and are logically inconsistent with F. The algorithm first extracts the set of all objects O that appear in F (step 1 in Algorithm 2), while keeping track of the type of each object. For example, suppose there are only two example worlds in F shown in Table 4.3. Then *extractObjects(F)* returns $\{P_1, P_2, P_7, P_8, C_3, C_5, 1, 2\}$.

[4]This is often referred to as the *covers* relation in inductive logic programming.

Example 1	**Example 2**
$\text{snap}(P_1, C_5, 1)$	$\text{snap}(P_7, C_3, 2)$
$\text{snap}(P_2, C_5, 1)$	$\text{snap}(P_8, C_3, 2)$
$\text{failedCapturing}(P_1, P_2, 1)$	$\text{failedFreeing}(P_2, P_5, 2)$

Table 4.3: Two simple examples of a logical representation a failed capture event.

In step 2, we limit ourselves to only hard formulas when testing compatibility. We do so since we can *prove* incompatibility only for hard formulas. Soft constraints can be violated many times in the data and yet we may not want to eliminate them. Instead, we want to merely adjust their weights, which is exactly what we do in our approach. Therefore, $\mathcal{T}_{\text{hard}}$ contains only hard formulas that appear in $\mathcal{T}$. Next, on lines 5 through 12, we check if the entire unmodified $\mathcal{T}_{\text{hard}}$ is compatible (since for $n = 0$, we do not remove any formulas). If it is compatible, we return an empty set indicating that all the hard formulas in the original seed theory $\mathcal{T}$ are compatible with the examples. If we detect incompatibility, we will need to remove some, and perhaps even all, hard formulas in order to arrive at a logically consistent theory. Therefore, we incrementally start removing n-tuples of formulas. That is, in the subsequent $|\mathcal{T}_{\text{hard}}|$ iterations of the while loop, we determine if we can restore consistency by removing any one of the hard formulas in $\mathcal{T}_{\text{hard}}$. If we can, we return the set $\mathcal{T}_{\text{hard}} \setminus f_i$, where f_i is the identified and removed incompatible formula. If consistency cannot be restored by removing a single formula, we in turn begin considering pairs of formulas ($n = 2$), triples ($n = 3$), *etc.* until we find a pruned theory $\mathcal{T}^c$ that is consistent with *all* examples.

In general, we do need to consider n-tuples of formulas, rather than testing each formula in isolation. This is due to disjunctive formulas in conjunction with an possibly incomplete truth assignment in the training data. Consider the following theory in

propositional logic:

$$f_1 = \neg a \vee b$$

$$f_2 = \neg b \vee c$$

$$\text{Data: } a \wedge \neg c$$

(Following the closed-world assumption, the negated atom c would actually not appear in the training data, but we explicitly include it in this example for clarity.) While f_1 and f_2 are each individually consistent with the data, $f_1 \wedge f_2$ is inconsistent with the data. More complicated examples can be constructed, where every group of k formulas is inconsistent with the data, even though the individual formulas are. In a special case where the truth values of all atoms in the training examples are known, the formulas can be tested for consistency individually, which reduces the original exponential number of iterations Algorithm 2 executes, in the worst case, to a linear complexity. An interesting direction for future work is to explore applications of logical methods to lower the computational cost for the general case of partially observed data.

We also note that some hard formulas model physical constraints or inviolable rules of capture the flag, and therefore hold *universally*. Appropriately, these formulas are not eliminated by Algorithm 2. As an example, consider formula H1 in Figure 4.5, which asserts that each player occupies exactly one cell at any given time. This formula is satisfied in games that include both successful and failed activities. On the other hand, consider formula H8 in the same figure. It contains a captured player to the cell he was captured in (following the "captured players cannot move" rule of CTF). While this holds for successful capturing events, it does not necessarily hold for failed attempts at capturing. Therefore, when rule H8 is expanded via second-order ML, only some of the derived formulas are going to be consistent with the observations.

Specifically, the candidate formula in Equation 4.1 will be pruned away, as it is inconsistent with the training examples, *i.e.*, players that were only nearly captured continue to be free to move about. However, the remaining three variants of formula H8 will not be pruned away. Equation 4.2 will always evaluate to true, since if someone attempts to re-capture an already captured player a, a does indeed remain stationary.

Similarly, Equation 4.3 is also consistent with all the example CTF games because if there is a failed attempt at capture immediately followed by a successful capture, the captured player does remain in place from time t onward. Finally, Equation 4.4 is compatible as well, since it is the original formula H8 that is consistent with the observations.

(4.1)
$$\forall a, t, c : \big(\text{isFailedCaptured}(a, t) \wedge \text{isFailedCaptured}(a, t{+}1) \wedge \text{snap}(a, c, t)\big) \Rightarrow \text{snap}(a, c, t{+}1)$$

(4.2)
$$\forall a, t, c : \big(\text{isCaptured}(a, t) \wedge \text{isFailedCaptured}(a, t+1) \wedge \text{snap}(a, c, t)\big) \Rightarrow \text{snap}(a, c, t+1)$$

(4.3)
$$\forall a, t, c : \big(\text{isFailedCaptured}(a, t) \wedge \text{isCaptured}(a, t+1) \wedge \text{snap}(a, c, t)\big) \Rightarrow \text{snap}(a, c, t+1)$$

(4.4) $\forall a, t, c : \big(\text{isCaptured}(a, t) \wedge \text{isCaptured}(a, t+1) \wedge \text{snap}(a, c, t)\big) \Rightarrow \text{snap}(a, c, t+1)$

The function *testSAT()* (line 11 in Algorithm 2) checks whether a given candidate theory $\mathcal{T}^c$ is compatible with the examples F by the following process. First, we ground $\mathcal{T}^c$ using the objects in O, thereby creating a ground theory $\mathcal{G}$. For example, if $\mathcal{T}^c = \{p(x) \Rightarrow q(x)\}$ and $O = \{B, W\}$, the grounding would be $\mathcal{G} = \{p(B) \Rightarrow q(B), p(W) \Rightarrow q(W)\}$. Then we check if $\mathcal{G} \cup F_{\text{hidden}}$ is satisfiable using the miniSAT solver, where F_{hidden} is simply the set of hidden atoms that appear in F. Intuitively, this corresponds to testing whether we can "plug in" the worlds in F into $\mathcal{T}^c$ while satisfying all the hard constraints. Though satisfiability is an NP-complete problem, in practice *testSAT()* completes within tenths of a second even for the largest problems in our CTF domain.

For instance, suppose $F_{\text{hidden}} = \{p(B), \neg q(B)\}$. Then we test satisfiability of the formula

$$\Big(p(B) \Rightarrow q(B)\Big) \wedge \Big(p(W) \Rightarrow q(W)\Big) \wedge p(B) \wedge \neg q(B).$$

In this case we cannot satisfy it since we are forced to set $p(B)$ to *true* and $q(B)$ to *false*, which renders the first clause—and therefore the whole formula—*false*.

An alternative approach to pruning formulas via satisfiability testing, as we have just described, would be to treat both types of formulas (hard and soft) in the inflated theory $\mathcal{M}'_S$ as strictly soft formulas and learning a weight for each formula from examples of both successful and failed game events. However, this introduces several complications that negatively impact the system's performance as well as model clarity. First, the number of formulas in the inflated theory can be exponentially larger than in the seed theory. While the instantiation of the second-order ML representation can be quantified to limit this expansion, we still have worst-case exponential blow-up. By treating all formulas as soft ones, we now need to potentially learn many more weights. This is especially problematic for activities that occur rarely, as we may not have enough training data to properly learn those weights. Eliminating the hard candidate formulas by proving them inconsistent dramatically reduces the number of parameters we have to model. While satisfiability testing is NP-complete, weight learning in Markov logic entails running inference multiple times, which is itself in general a #P-complete problem.

The second reason for distinguishing between soft and hard formulas is the resulting clarity and elegance of the final learned model $\mathcal{M}_{S+F}$. Even in situations when we have enough training data to properly learn a large number of weights, we run into overfitting problems, where neither the structure nor the parameters of the model represent the domain in a natural way. Our experiments have shown that if we skip the pruning stage (steps 3 and 4 in Algorithm 1), the model's recognition performance does not differ from that of a pruned model in a significant way (p-value of 0.45). However, we end up with a large number of soft formulas with a mixture of positive and negative weights that the learning algorithm carefully tuned and balanced to fit the training data. They however bear little relationship to the concepts in the underlying domain. Not only does this make it very hard for a human expert to analyze the model, but it makes it even harder to modify the model.

For these reasons, softening all hard formulas is, in general, infeasible. An interesting

direction of future work will be to identify a small amount of *key* inconsistent hard formulas to soften, while eliminating the rest of the inconsistent hard formulas. This however entails searching in a large space of candidate subsets of softened formulas, where each iteration requires expensive re-learning of all weights.

Note that Algorithm 2 terminates as soon as it finds a compatible theory that requires the smallest number of formula-removals. We also experimented with an active learning component to our system, where we modify Algorithms 1 and 2 such that they present *several* possible refinements of the theory to the user who then selects the one that looks best. The proposed modifications are shown both at the ML theory level with modified sections (formulas) highlighted as well as at the data level where the program shows the inferred consequences of those modifications. For each candidate modification, the corresponding consequences are displayed as a collection of animations where each animation shows what the results of activity recognition would be if we committed to that particular candidate theory. Note that even people who do not have background in ML can interact with such a system since the visualization is easy to understand. Interestingly, in the case of captures and frees, the least modified theory that the "off-line" version of the algorithm finds is also the best one and therefore there is no need to query the user. One can view this as a differential variant of Occam's razor. However, for different activities or other domains, the active learning approach may be worth revisiting and we leave its exploration for future work.

Finally, general structure learning techniques from statistical-relational AI and from inductive logic programming are not applicable as a substitute for our theory augmentation algorithm for several reasons. The main reason is that, for efficiency reasons, existing techniques in the literature typically operate over a very restricted set of formula templates. That is, they consider only Horn clauses, or only formulas without an existential quantifier, or only formulas with at most k literals or with at most l variables, and so on. This set of restrictions is part of the language bias of any given approach. While in principle, structure learning is possible without a language bias, one often has to carefully define one for the sake of tractability (see the Section 4.7 for details). In our approach, the language bias is defined implicitly as discussed in Section 4.5.2.

4.5.3 Extracting The Goal From Success and Failure

Recall that applying the theory augmentation process (Algorithm 1) on the CTF seed theory of successful interactions (shown in Figures 4.5 and 4.6) induces a new set of formulas that capture the structure of failed activities and ties them together with the existing formulas in the seed theory.

The logically inconsistent formulas $\mathcal{I}$ that Algorithm 2 returns are ones that are not satisfiable in the worlds with failed activities. At the same time, variants of those formulas were consistent with the examples of successful actions occurring in the games. Therefore, $\mathcal{I}$ represents the *difference* between a theory that models only successful activities and the augmented theory of both successful and failed actions, that has been derived from it. Intuitively, the difference between success and failure can be viewed as the intended purpose of any given activity a rational agent executes, and consequently as the goal the agent has in mind when he engages in that particular activity. In the next section, we will explore the goals extracted from the CTF domain in this fashion.

This concludes discussion of our models and methodology, and now we turn to experimental evaluation of the framework presented above.

4.6 Experiments and Results

We evaluate our approach along the three major directions outlined in Section 4.5 (Methodology), while focusing on answering the four research questions formulated *ibidem*. The structure of this section closely follows that of the Methodology section.

In a nutshell, we are first interested in how our Markov logic models perform on the standard multi-agent activity recognition task—labeling successful activities—and how their performance compares to the alternative models. Second, we examine the augmented model that captures both successful and failed attempts at activities. This is the model $\mathcal{M}_{S+F}$ induced by Algorithm 1, which also lets us extract the intended goal of the activities in question. Third, we compare the performance of $\mathcal{M}_{S+F}$ on the task of jointly recognizing *all four* activities with that of an alternative model. Finally, we

investigate to what extent the reasoning about failed attempts does help in recognition of successfully executed activities.

All experiments are performed on our capture the flag dataset consisting of four separate games. The dataset is summarized in Table 4.4, where for each game we list the number of raw GPS readings and the number of instances of each activity of interest. We evaluate the models via four-fold cross-validation, always training on three games (if training is required for a model) and testing against the fourth. For each experimental condition below, we report precision, recall, and F1 scores attained by each respective model over the four cross-validation runs. We have purposefully chosen to split the data so that each cross-validation fold directly corresponds to a separate game of CTF for conceptual convenience and clarity. As we discussed above, the events occurring in the games often have far-reaching consequences. For example, most captured players are never freed by their allies. Therefore, a capture at the beginning of a game typically profoundly influences the entire rest of the game. For this reason, splitting the games randomly or even manually would introduce unnecessary complications, as most of the segments would have dependencies on other segments. By enforcing that each fold exactly corresponds with a different game, we make each fold self-contained.

To quantify the statistical significance of the pair-wise differences between models, we use a generalized probabilistic interpretation of F1 score (Goutte and Gaussier, 2005). Namely, we express F1 scores in terms of gamma variates derived from models' true positives, false positives, and false negatives (Goutte and Gaussier, 2005, $\lambda = 0.5$, $h = 1.0$, *cf.*,). This approach makes it possible to compare our results to future work that may apply alternative models on similar, but not identical, datasets. A future comparison may, for instance, include additional games or introduce random splits of the data. We note that standard statistical significance tests cannot be applied in those situations. All p-values reported are one sided, as we are interested if models' performance significantly *improves* as their level of sophistication increases.

	#GPS	#AC	#FC	#AF	#FF
Game 1	13,412	2	15	2	1
Game 2	14,420	2	34	2	1
Game 3	3,472	6	12	0	2
Game 4	10,850	3	4	1	0
Total	42,154	13	65	5	4

Table 4.4: CTF dataset overview: #GPS is the total number of raw GPS readings, #AC and #FC is the number actual (successful) and failed captures respectively, and analogously for freeings (#AF and #FF).

4.6.1 Recognition of Successful Activities

Recall that for both our two-step (2SML) and unified (UML) Markov logic models, we specify the Markov logic formulas by hand and optimize the weights of the soft formulas via supervised on-line learning. We run a modified version of theBeast software package to perform weight learning and MAP inference. theBeast implements the cutting plane meta solving scheme for inference in Markov logic, where the ground ML network is reduced to an integer linear program that is subsequently solved by the LpSolve ILP solver. We chose this approach as opposed to, *e.g.*, MaxWalkSAT that can get "stuck" at a local optimum, since the resulting run times are still relatively short (under an hour even for training and testing even the most complex model).

At weight learning time, we use the margin infused relaxed algorithm (MIRA) for weight updates while the loss function is computed from the number of false positives and false negatives over the hidden atoms, as described in the Methodology section. The discretization step for the dynamic Bayesian network model (DBN) is implemented in Markov logic and is also executed in this fashion. The DBN model is trained via maximum likelihood as described in Section 3.2.1. The two deterministic baselines (B and B+S) do not require any training phase.

At inference time, we are interested in the most likely explanation of the data. In

Markov logic, maximum *a posteriori* inference reduces to finding a complete truth assignment that satisfies all the hard constraints while maximizing the sum of the weights of the satisfied soft formulas. At testing time, theBeast Markov logic solver finds the most likely truth assignment to the hidden atoms as described above, and in this section we are specifically interested in the values of the capturing and freeing atoms.

In DBNs, the most likely explanation of the observations is equivalent to Viterbi decoding. The DBN model assigns either free or captured state to each player for every time step. We then label all transitions from free to captured state as capturing and all transitions from captured to free as freeing. Note that the DBN model is capable of determining which player is being freed or captured, but it does not model which player does the freeing or capturing. In our evaluation, we give it the benefit of the doubt and assume it always outputs the correct actor.

For all models, inference is done simultaneously over an entire game (on average, about 10 minutes worth of data). Note that we do not restrict inference to a (small) sliding time window. As the experiments described below show, many events in this domain can only be definitely recognized long after they occur. For example, GPS noise may make it impossible to determine whether a player has been captured at the moment of encounter with an enemy, but as the player thereafter remains in place for a long time, the possibility of his capture becomes certain.

Figures 4.7 and 4.8 summarize the performance of our models of successful capturing and freeing in terms of precision, recall, and F1 score calculated over the four cross-validation runs. For clarity, we present the results in two separate plots, but each model was jointly labeling both capturing and freeing activities. We do not consider the baseline model for freeing recognition as that activity makes little sense without having a notion of player state (captured or free).

We see that the unified approach yields the best results for both activities. Let us focus on capturing first (Figure 4.7). Overall, the unified model labels 11 out of 13 captures correctly—there are only two false negatives. In fact, these two capture events are missed by *all* the models because they involve two enemies that appear unusually far apart (about 12 meters) in the raw data. Even the unified approach fails on this

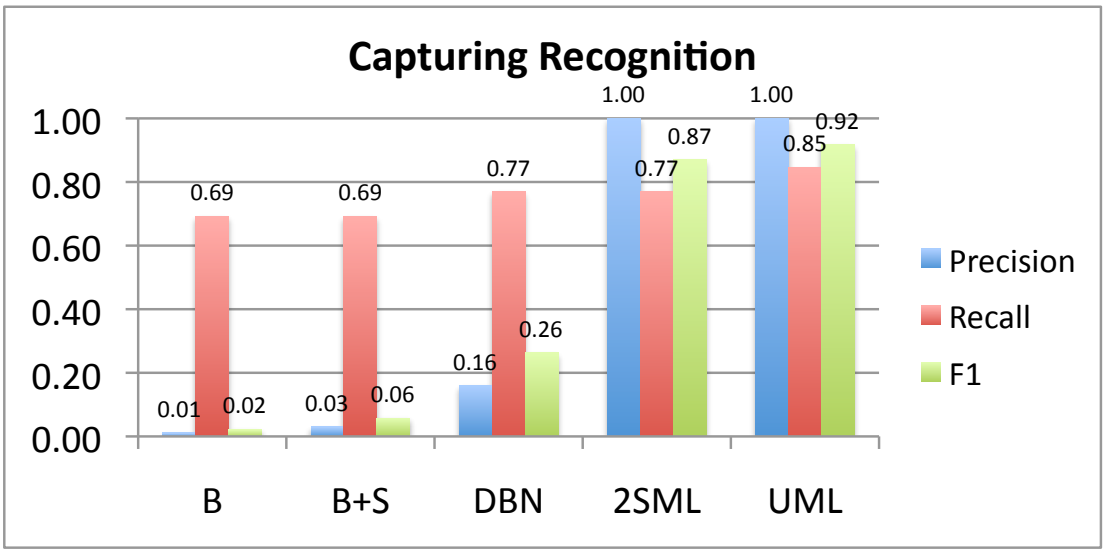


Figure 4.7: Comparison of performance of the five models on capturing recognition while doing joint inference over both capturing and freeing events. See Table 4.5 for statistical significance analysis of the pairwise differences between models. (B = baseline model, B+S = baseline model with states, 2SML = two-step Markov logic model, UML = unified Markov logic model)

instance since the cost of adjusting the players' trajectories—thereby losing score due to violation of the geometry-based constraints—is not compensated for by the potential gain from labeling an additional capture.

Note that even the two-step approach recognizes 10 out of 13 captures. As compared to the unified model, it misses one additional instance in which the involved players, being moderately far apart, are snapped to mutually nonadjacent cells. On the other hand, the unified model does not fail in this situation because it is not limited by prior nonrelational snapping to a few nearby cells. However, the difference between their performance on our dataset is not statistically significant even at the 0.05 level (p-value of 0.32).

Both deterministic baseline models (B and B+S) perform very poorly. Although they yield a respectable recall, they produce an overwhelming amount of false positives. This shows that even relatively comprehensive pattern matching does not work at all

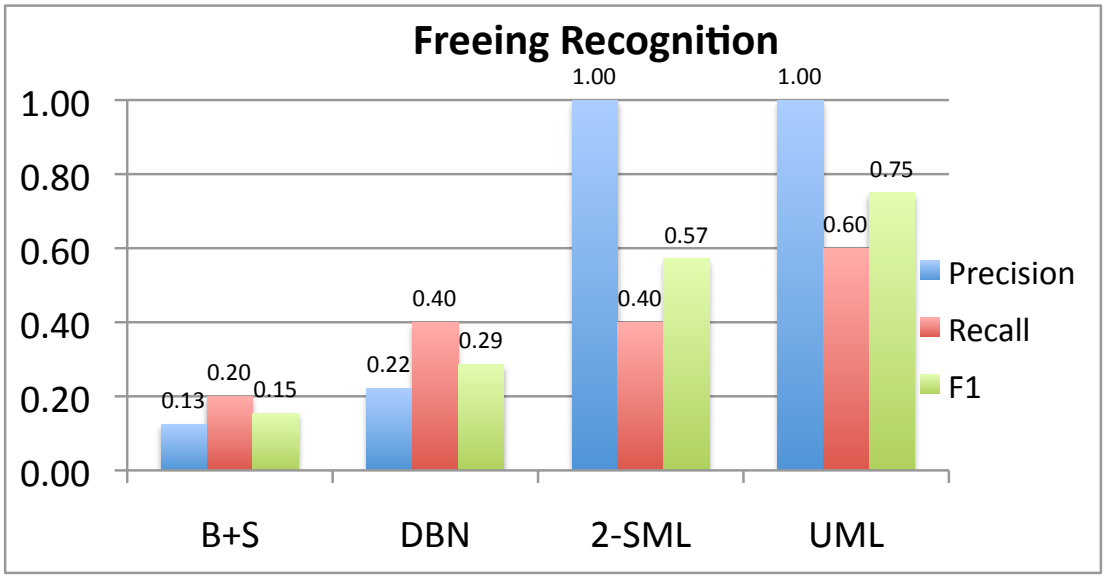


Figure 4.8: Comparison of performance of our three models on freeing recognition while doing joint inference over both capturing and freeing events. See Table 4.6 for statistical significance analysis of the pairwise differences between models. (B+S = baseline model with states, 2SML = two-step Markov logic model, UML = unified Markov logic model)

in this domain. Interestingly, the performance of the DBN model leaves much to be desired as well, especially in terms of precision. While the DBN model is significantly better than both baselines (p-value less than 5.9×10^{-5}), it also achieves significantly worse performance than both the Markov logic models (p-value less than 0.0002; see Table 4.5).

Table 4.5 summarizes p-values of pairwise differences between models of actual (*i.e.*, successful) capturing. While the difference between the Markov logic-based models (2SML and UML) are not statistically significant (p-value of 0.32), pairwise differences in F1 scores between all other models are significant at the 0.02 level, and most often even at much lower p-values.

Though the unified model still outperforms its alternatives in the case of freeing recognition as well, its performance is further from ideal as compared to the capture recognition case (Figure 4.8). It correctly identifies only 3 out of 5 freeing events in the games, but does not produce any false positives. This is partly due to the dependency of

	B+S	**DBN**	**2SML**	**UML**
B	0.0192	3.6×10^{-6}	5.1×10^{-7}	2.9×10^{-7}
B+S	-	5.9×10^{-5}	9.4×10^{-6}	1.4×10^{-6}
DBN	-	-	0.0002	8.0×10^{-5}
2SML	-	-	-	0.3230

Table 4.5: Summary of statistical significance (one sided p-values) of the pairwise differences between F1 scores for models of actual capturing. (B = baseline model, B+S = baseline model with states, DBN = dynamic Bayesian network model, 2SML = two-step Markov logic model, UML = unified Markov logic model)

	DBN	**2SML**	**UML**
B+S	0.2739	0.0733	0.0162
DBN	-	0.1672	0.0497
2SML	-	-	0.2743

Table 4.6: Summary of statistical significance (one sided p-values) of the pairwise differences between F1 scores for models of actual freeing. (B+S = baseline model with states, DBN = dynamic Bayesian network model, 2SML = two-step Markov logic model, UML = unified Markov logic model)

freeing on capturing. A failure of a model to recognize a capture precludes its recognition of a future freeing. Another reason is the extreme sparseness of the freeing events (there are only five of them in 40,000+ datapoints). Finally, in some instances players barely move after they had been freed. This may occur for a number of reasons ranging from already occupying a strategic spot to simply being tired. Such freeing instances are very challenging for any automated system, and even people familiar with the game to recognize (several situations would have been extremely hard to disambiguate if we didn't have access to our notes about data collection).

The two-step ML model does a slightly worse job than the unified model on freeing recognition. It correctly identifies only 2 out of 5 freeings for the same reasons as in the capturing recognition case. Similarly to models of actual captures, the difference between the unified and two-step freeing models is not statistically significant (p-value of 0.27).

Table 4.6 summarizes p-values of pairwise differences between models of actual (*i.e.*, successful) freeing. Here we see that only the difference between B+S and UML models is statistically significant (p-value of 0.01), whereas the differences between the rest of the model pairs are not statistically significant. Since there are only five instances of successful freeing, the 2SML model does not perform significantly better than the B+S model at the 0.05 significance level (p-value of 0.07). However, the UML model achieves better recognition results than even the DBN model with high confidence (p-value less than 0.05). Therefore, we see that although the 2SML model strictly dominates the non-Markov logic models when evaluated on capturing recognition, we need the full power of the unified ML model to strictly outperform the nonrelational alternatives for freeing. This suggests that as we move to more complex and more interdependent activities, relational and unified modeling approaches will be winning by larger and larger margins.

Even though the statistical significance tests suggest that 2SML is likely to give similar results to UML, it is important to note that 2SML, by design, *precludes* recognition of the activities in question in certain situations. Namely, as our experiments demonstrate, when the players are snapped to cells that are too far apart, the two-step

model does not even consider those instances as candidates for labeling, and inevitably fails at recognizing them. Therefore, one needs to look beyond the p-values obtained when comparing the fully unified models to various alternatives.

As expected from the experiments with capturing recognition, both deterministic baseline models perform very poorly on freeing recognition as well. Not only do they produce an overwhelming amount of false positives, they also fail to recognize most of the freeing events.

Thus, we see that the models cast in Markov logic perform significantly better than both of the deterministic baseline models, and also better than the probabilistic, but nonrelational, DBN model. We note that the DBN model has the potential to be quite powerful and similar DBNs have been applied with great success in previous work on activity recognition from location data (Eagle and Pentland, 2006; Liao *et al.*, 2007b). It also has many similarities with the two-step ML model. They both share the same denoising and discretization step, and they both operate on the same observed data. The key difference is that the DBN model considers players individually, whereas the two-step ML model performs joint reasoning.

Looking at the actual CTF game data, we see several concrete examples of how this hurts DBN's labeling accuracy. For instance, consider a situation where two allies had been captured near each other. Performing inference about individual players in isolation allows the DBN model to infer that the two players effectively free each other, even though in reality they are both captured and cannot do so. This occurs because the DBN model is oblivious to the explicit states of one's teammates as well as opponents. Since capturing and freeing are interdependent, the obliviousness of the DBN model to the state of the actors negatively impacts its recognition performance for both activities. The example we just gave illustrates one type of freeing false positives. The hallucinated freeings create opportunities that often lead to false positives of captures, creating a vicious cycle. False negatives of freeing (capturing) events often occur for players who the model incorrectly believes have already been freed (captured) at a prior time.

Since the Markov logic based models are significantly better—with a high level of confidence—than the alternatives that are not fully relational, the experiments above

validate our hypothesis that we need to exploit the rich relational and temporal structure of the domain in a probabilistic way and at the same time affirmatively answer research question Q1 (*Can we reliably recognize complex multi-agent activities in the CTF dataset even in the presence of severe noise?*). Namely, we show that although relatively powerful probabilistic models are not sufficient to achieve high labeling accuracy, we can gain significant improvements by formulating the recognition problem as learning and inference in Markov logic networks.

Now we turn to the evaluation of our method of learning models of both success and failure in people's activities.

4.6.2 Learned Formulas and Intentions

Applying the theory augmentation process (Algorithm 1) on the CTF seed theory (shown in Figures 4.5 and 4.6) induces a new set of formulas that capture the structure of failed activities and ties them together with the existing formulas in the theory. We call this model $\mathcal{M}_{S+F}$. Figure 4.9 shows examples of new weighted formulas modeling failed freeing and capturing attempts that appear in $\mathcal{M}_{S+F}$.

First, note that our system correctly carries over the basic preconditions of each activity (contrast formulas S4 with S4′ and S5 with S5′ in Figures 4.6 and 4.9 respectively). This allows it to *reliably* recognize both successful and failed actions instead of, *e.g.*, merely labeling all events that at some point in time appear to resemble a capture as near-capture. This re-use of preconditions directly follows from the language bias of the theory augmentation algorithm.

Turning our attention to the learned hard formulas, we observe that the system correctly induced equivalence classes of the states, and also derived their mutual exclusion relationships (H5′). It furthermore tied the new failure states to their corresponding instantaneous interactions (H6′ and H7′).

Finally, the algorithm correctly discovers that the rule *"If a player is captured then he or she must remain in the same location"* (H8, Figure 4.5) is the key distinction between a successful and failed capture (since players who were not actually captured

can still move). Therefore, it introduces an appropriate rule for the failed captures (H8′, Figure 4.9) explicitly stating that failed capturing does not confine the near-captured player to remain in stationary. An analogous process yields a fitting separation between failed and successful freeings. Namely, our model learns that an unsuccessfully freed player remains stationary. This learned difference between success and failure in players' actions directly corresponds to the goal of the activity and consequently the intent of rational actors. This difference is what our system outputs as the intended goal of capturing activity (and analogously for freeing).

These experimental results provide an evidence for a resounding "yes" to both Q2 (*Can models of attempted activities be automatically learned by leveraging existing models of successfully performed actions?*) and Q3 (*Does modeling both success and failure allow us to infer the respective goals of the activities?*) within the CTF domain.

We note that instead of applying our automated theory augmentation method, a person could, in principle, manually formulate a Markov logic theory of successful as well as failed activities by observing the games. After all, this is how we designed the initial seed model of successful events. However, this process is extremely time consuming, as one tends to omit encoding facts that to us, humans, seem self-evident but need to be explicitly articulated for the machine (*e.g.*, a single person cannot be at ten different places at once, or that a player is either free or captured but not both). It is also surprisingly easy to introduce errors in the theory, that are difficult to debug, mostly because of the complex weight learning techniques involved. Therefore, we believe that the theory augmentation method is a significant step forward in enhancing models' capabilities while requiring small amounts of human effort. As the complexity of domains and their models increases, this advantage will gain larger and larger importance.

4.6.3 Recognition of Both Successful and Failed Activities

We now compare the performance of our model $\mathcal{M}_{S+F}$ to an alternative (baseline) method that labels all four activities in the following way. Similarly to the baseline with states model for successful interactions defined in Section 4.5.1, there are two separate stages. First we snap each GPS reading to the nearest cell by applying only

(S4′) $\forall a_1, a_2, t : [(\text{enemies}(a_1, a_2) \wedge \text{onHomeTer}(a_1, t) \wedge$
$\text{onEnemyTer}(a_2, t) \wedge \text{samePlace}(a_1, a_2, t) \wedge \text{isFree}(a_1, t)$
$\wedge \text{isFree}(a_2, t)) \Rightarrow \text{failedCapturing}(a_1, a_2, t)] \cdot 11.206$

(S5′) $\forall a_1, a_2, t : [(\neg\text{enemies}(a_1, a_2) \wedge \text{onEnemyTer}(a_1, t) \wedge$
$\text{onEnemyTer}(a_2, t) \wedge \text{samePlace}(a_1, a_2, t) \wedge \text{isFree}(a_1, t)$
$\wedge \text{isCaptured}(a_2, t)) \Rightarrow \text{failedFreeing}(a_1, a_2, t)] \cdot 1.483$

(S6′) $\forall a_1, a_2, t : [\text{failedCapturing}(a_1, a_2, t)] \cdot (-0.0001)$

(S7′) $\forall a_1, a_2, t : [\text{failedFreeing}(a_1, a_2, t)] \cdot (-0.002)$

(H5′) $\neg\forall a, t : \text{isFailedCaptured}(a, t) \oplus \text{isFree}(a, t)$
$\neg\forall a, t : \text{isCaptured}(a, t) \oplus \text{isFailedFree}(a, t)$
$\forall a, t : \text{isFailedCaptured}(a, t) \Leftrightarrow \text{isFree}(a, t)$
$\forall a, t : \text{isCaptured}(a, t) \Leftrightarrow \text{isFailedFree}(a, t)$

(H6′) $\forall a, t : (\text{isFree}(a, t) \wedge \text{isFailedCaptured}(a, t+1)) \Rightarrow (\exists_{=1} a_1 : \text{failedCapturing}(a_1, a, t))$

(H7′) $\forall a, t : (\text{isCaptured}(a, t) \wedge \text{isFailedFree}(a, t+1)) \Rightarrow (\exists_{=1} a_1 : \text{failedFreeing}(a_1, a, t))$

(**H8′**)
$\neg\forall \mathbf{a}, \mathbf{t}, \mathbf{c} : (\mathbf{isFailedCaptured}(\mathbf{a}, \mathbf{t}) \wedge \mathbf{isFailedCaptured}(\mathbf{a}, \mathbf{t+1}) \wedge \mathbf{snap}(\mathbf{a}, \mathbf{c}, \mathbf{t})) \Rightarrow \mathbf{snap}(\mathbf{a}, \mathbf{c}, \mathbf{t+1})$

Figure 4.9: Example formulas, learned by Algorithm 1, that model unsuccessful capturing and freeing events. The crucial intent recognition formula (H8′) is highlighted in bold. Formulas eliminated by Algorithm 2 are preceded by the ¬ symbol, and are not included in the induced model $\mathcal{M}_{S+F}$. The identity $\text{isCaptured}(a, t) = \neg\text{isFree}(a, t)$ is applied throughout refining to show the formulas in a more intuitive fashion. For concreteness sake, the values of the learned weights here come from one cross-validation run (and are similar in other runs).

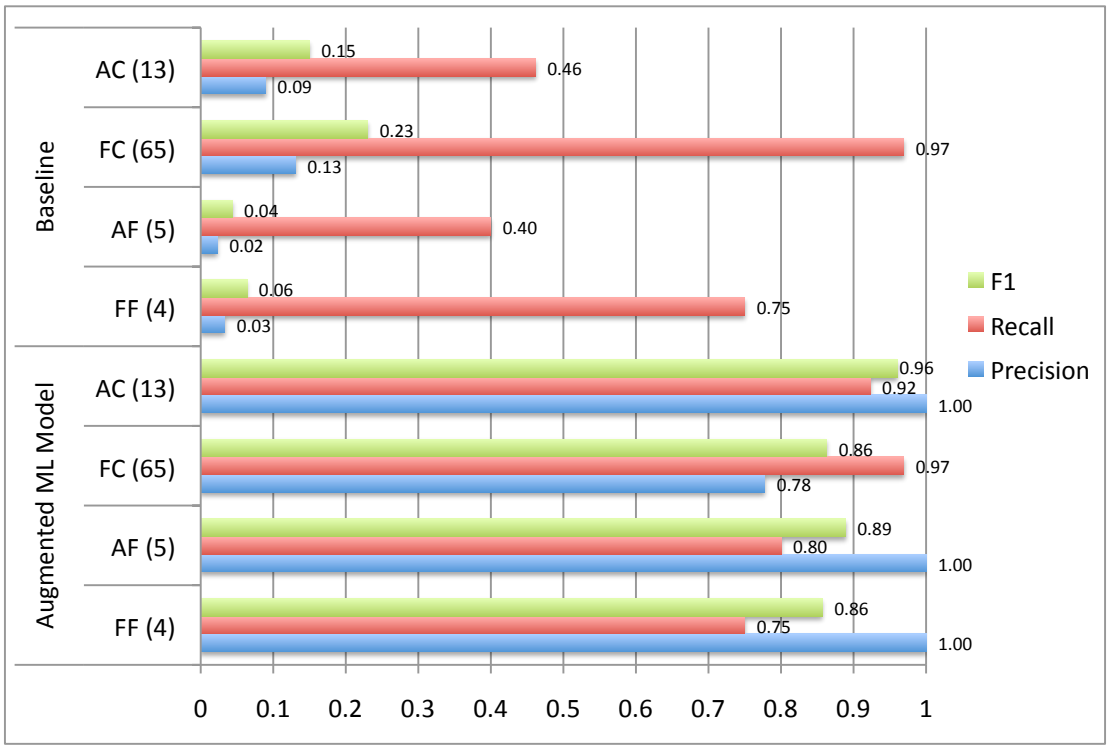


Figure 4.10: Performance of the baseline and augmented ($\mathcal{M}_{S+F}$) models on joint recognition of successful and failed capturing and freeing. The F1 score of the augmented model is significantly better than that of the baseline for all four target activities (p-value less than 1.3×10^{-4}). AC = actual (successful) capturing, FC = failed capturing, AF = actual freeing, FF = failed freeing.

the geometric constraints (H1 and S1–S3) of our theory, and afterward we label the instances of our activities. The following labeling rule is applied. We loop over the whole discretized (via snapping) data set and look for instances where a pair of players a and b were snapped (in the first step) to either the same cell or to two adjacent cells at time t, they are enemies, b is not captured already, and a is on its home territory while b is not. If b moves (is snapped to a different cell at a later time) *without* having an ally nearby, we output *failedCapturing(a,b,t)*, otherwise we output *capturing(a,b,t)*. The labeling rule for freeing is defined analogously and all four events are tied together. We also tested a variant of the DBN model introduced in Section 4.5.1 that has two additional hidden state values for node S_t: isFailedFree and isFailedCaptured. However, the difference in the results obtained with this model was not statistically significant (p-value of 0.38), and therefore we focus on the conceptually more straightforward baseline model described above.

Model $\mathcal{M}_{S+F}$ is evaluated using four-fold cross-validation (always training on three games and testing against the fourth). Figure 4.10 compares both models in terms of precision, recall, and F1 score. Note that all four activities are modeled *jointly* in both models. The F1 score of the augmented model is significantly better than that of the baseline for all four target activities (p-value less than 1.3×10^{-4}).

We see that the baseline model has, in general, a respectable recall but it produces a large number of false positives for all activities. The false positives stem from the fact that the algorithm is "greedy" in that it typically labels a situation where several players appear close to each other for certain period of time as a sequence of many captures and subsequent frees even though none of them actually occurred. Model $\mathcal{M}_{S+F}$ gives significantly better results because it takes full advantage of the structure of the game in a probabilistic fashion. It has a similar "over labeling" tendency only in the case of failed captures, where a single capture attempt is often labeled as several consecutive attempts. While this hurts the precision score, it is not a significant deficiency, as in practice, having a small number of short game segments labeled as possible near-captures is useful as well.

We also note that even though the original model (UML) did not contain any infor-

mation on failed capturing nor failed freeing, the performance of $\mathcal{M}_{S+F}$ is respectable even for those two newly introduced activities. We only provided examples of game situations where those attempts occur and the system augmented itself and subsequently labeled all four activities. Thus, we see that we can indeed extend preexisting models in an automated fashion so that the unified model is capable of recognizing not only individual activities, but also both success and failure in people's behavior.

4.6.4 The Effect of Modeling Failed Attempts on Recognition of Successful Activities

To address research question Q4 (*Does modeling failed attempts of activities improve the performance on recognizing the activities themselves?*), we want to see how much does the recognition of attempted activities help in modeling the successful actions (the latter being the standard activity recognition problem). Toward that end, we compare the Markov logic model $\mathcal{M}_S$ that jointly labels only successful capturing and freeing with model $\mathcal{M}_{S+F}$ that jointly labels both successful and failed attempts at both capturing and freeing (see Section 4.5.2 for a detailed description of the two models). However, we evaluate them in terms of precision, recall, and F1 score only on *successful* interactions, not all four types of activities.

Figure 4.11 summarizes the results. We see that when evaluated on actual capturing, $\mathcal{M}_{S+F}$ performs better than $\mathcal{M}_S$, and similarly for freeing. However, the difference in F1 scores between a model that captures both attempted and successful activities ($\mathcal{M}_{S+F}$) and a model of only successful activities ($\mathcal{M}_S$) is not statistically significant (p-value of 0.20). This is partly because $\mathcal{M}_S$ already produces very solid results, leaving little room for improvement. Additionally, the CTF dataset contains relatively few events of interest. In terms of labeling performance at testing time, the difference between the two models is more than 11% ($\mathcal{M}_S$ and $\mathcal{M}_{S+F}$ recognize, respectively, 14 and 16 out of 18 successful activities correctly). Thus, we believe the trends shown in Figure 4.11 are promising and modeling attempted actions does improve recognition performance on both capturing and freeing, but evaluation on a dataset with a larger number of events is needed to show the difference to be statistically significant at a higher confidence

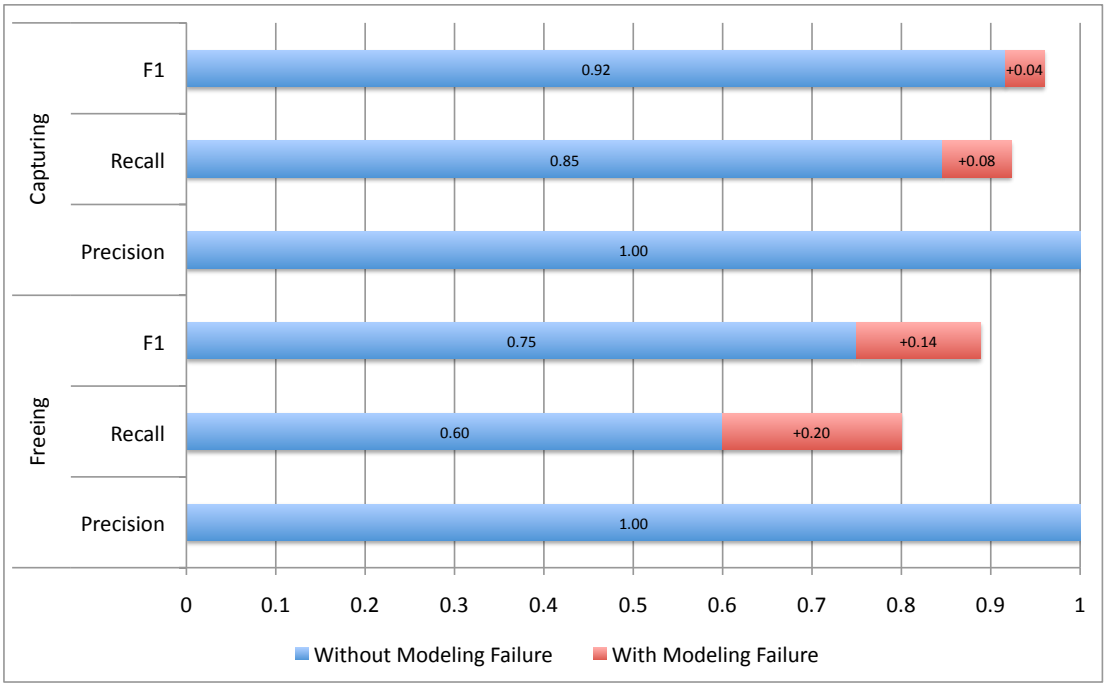


Figure 4.11: Considering unsuccessfully attempted activities strictly improves performance on standard activity recognition. Blue bars show scores obtained with the unified Markov logic model that considers only *successful* activities ($\mathcal{M}_S$). The red bars indicate the additive improvement provided by the augmented model that considers both *successful and failed* activities ($\mathcal{M}_{S+F}$, the output of Algorithm 1). Each model labels its target activities jointly, we separate capturing and freeing in the plot for clarity. Precision has value of 1 for both models. F1 scores obtained when explicitly modeling failed attempts are not statistically different from F1 scores obtained without modeling attempts at a high confidence level (p-value of 0.20). However, these results still show the importance of reasoning about people's attempts when recognizing their activities; see text for details.

level. However, this does not mean that recognizing attempts is unimportant. As we show above, our induced augmented model does recognize failed (as well as successful) activities in the complex CTF domain with high accuracy, and we argue this to be a significant contribution.

Finally, the comparison of $\mathcal{M}_S$ and $\mathcal{M}_{S+F}$ shows that applying our learning algorithm that augments a model with more recognition capabilities *does not hurt* model labeling performance. The fact that binary classification problems are typically easier to solve than their multi-class counterparts has been well reported on in machine learning literature (Allwein *et al.*, 2001). Therefore, introducing new activities into a model, especially in an automated way, is likely to degrade its performance. Contrary to this intuition, our experiments show that $\mathcal{M}_{S+F}$ is no worse than $\mathcal{M}_S$ on successful activity recognition (*i.e.*, their intersection) with high confidence, even though $\mathcal{M}_{S+F}$ is clearly richer and more useful.

4.7 Related Work

In the world of *single-agent* location-based reasoning, the work of Bui (2003) presents and evaluates a system for probabilistic plan recognition cast as an abstract hidden Markov memory model. Subsequently, the work of Liao *et al.* (2004) implements a system for denoising raw GPS traces and simultaneously inferring individuals' mode of transportation (car, bus, *etc.*) and their goal destination. They cast the problem as learning and inference in a dynamic Bayesian network and achieve encouraging results. In a follow-up work, Liao *et al.* (2005) introduce a framework for location-based activity recognition, which is implemented as efficient learning and inference in a relational Markov network.

The work of Ashbrook and Starner (2003) focuses on inferring significant locations from raw GPS logs via clustering. The transition probabilities between important places are subsequently used for a number of user modeling tasks, including location prediction. The work of Eagle and Pentland (2006) explores harnessing data collected on regular smart phones for modeling human behavior. Specifically, they infer individuals' general

location from nearby cell towers and Bluetooth devices at various times of day. Applying a hidden Markov model (HMM), they show that predicting if a person is at home, at work, or someplace else can be achieved with more than 90% accuracy. Similarly, the work of Eagle and Pentland (2009) extracts significant patterns and signatures in people's movement by applying eigenanalysis to smart phone logs.

The work of Hu *et al.* (2008) concentrates on recognition of interleaving and overlapping activities. They show that publicly available academic datasets contain a significant number of instances of such activities, and formulate a conditional random field (CRF) model that is capable of detecting them with high (more than 80%) accuracy. However, they focus solely on single-agent household activities.

People's conversation has been the primary focus of *multi-agent* modeling effort (Barbuceanu and Fox, 1995). In the fields of multi-agent activity recognition and studies of human behavior, researchers have either modeled conversation explicitly (Busetta *et al.*, 2001, *e.g.*,), or have leveraged people's communication implicitly via call and location logs from mobile phones. This data has been successfully used to infer social networks, user mobility patterns, model socially significant locations and their dynamics, and others (Eagle and Pentland, 2006; Eagle *et al.*, 2009). This is arguably an excellent stepping stone for full-fledged multi-agent activity recognition since location is, at times, practically synonymous with one's activity (*e.g.*, being at a store often implies shopping) (Tang *et al.*, 2010), and our social networks have tremendous influence on our behavior (Pentland, 2008).

Additionally, a number of researchers in machine vision have worked on the problem of recognizing events in videos of sporting events, such as impressive recent work on learning models of baseball plays (Gupta *et al.*, 2009). Most work in that area has focused on recognizing individual actions (*e.g.*, catching and throwing), and the state of the art is just beginning to consider relational actions (*e.g.*, the ball is thrown from player A to player B). The computational challenges of dealing with video data make it necessary to limit the time windows of a few seconds. By contrast, we demonstrate in this work that many events in the capture the flag data can only be disambiguated by considering arbitrarily long temporal sequences. In general, however, both our work

and that in machine vision rely upon similar probabilistic models, and there is already some evidence that statistical-relational techniques similar to Markov logic can be used for activity recognition from video (Biswas *et al.*, 2007; Tran and Davis, 2008).

Looking beyond activity recognition, recent work on relational spatial reasoning includes an attempt to locate—using spatial abduction—caches of weapons in Iraq based on information about attacks in that area (Shakarian *et al.*, 2009). Additionally, the work of Abowd *et al.* (1997) presents a location- and context-aware system, Cyberguide, that helps people explore and fully experience foreign locations. Other researchers explore an intelligent and nonintrusive navigation system that takes advantage of predictions of traffic conditions along with a model of user's knowledge and competence (Horvitz *et al.*, 2005). Finally, the work of Kamar and Horvitz (2009) explore automatic generation of synergistic plans regarding sharing vehicles across multiple commuters.

An interesting line of work in cognitive science focuses on intent and goal recognition in a probabilistic framework (Baker *et al.*, 2006, 2007). Specifically, they cast goal inference as inverse planning problem in Markov decision processes, where Bayesian inversion is used to estimate the posterior distribution over possible goals. Recent extensions of this work begin to consider simulated multi-agent domains (Baker *et al.*, 2008; Ullman *et al.*, 2010; Baker *et al.*, 2011). Comparison of the computational models against human judgement in synthetic domains shows a strong correlation between people's predicted and actual behavior. However, the computational challenges involved in dealing with the underlying partially observable Markov decision processes are prohibitive in more complex domains with large state spaces, such as ours.

The focus of our work is on a different aspect of reasoning about people's goals. Rather than inferring a distribution over possible, *a priori* known goals, we automatically *induce* the goals of complex multi-agent activities themselves.

Other researchers have concentrated on modeling behavior of people and general agents as reinforcement learning problems in both single-agent and multi-agent settings. The work of Ma (2008) proposes a system for household activity recognition cast as a single-agent Markov decision process problem that is subsequently solved using a probabilistic model checker. Wilson and colleagues address the problem of learning

agents' *roles* in a multi-agent domain derived from a real-time strategy computer game (Wilson *et al.*, 2008, 2010). Experiments in this synthetic domain show strongly encouraging results. While we do not perform role learning ourselves, we anticipate that the work of Wilson et al. is going to play an important role in learning hierarchies of people's activities. In our capture the flag domain, one can imagine automatically identifying a particular player as, for example, a defender and subsequently leveraging this information to model his or her behavior in a more "personalized" way.

The work of Hong (2001) concentrates on recognizing the goal of an agent in the course of her activities in a deterministic, but relational setting. Interesting work on goal recognition has been also applied to computer-aided monitoring of complex multi-agent systems, where relationships between agents are leveraged to compensate for noise and sparse data (Kaminka *et al.*, 2002). By contrast, in our work we focus on *learning* the respective goals of a given set of multi-agent activities in a probabilistic setting. The knowledge is in turn leveraged to achieve a stronger robustness of the other recognition tasks. Similarly to the approach of Hong, our system does not need a supplied plan library either.

Our work also touches on anomaly detection since our system reasons about the failed attempts of the players. Anomaly detection concerns itself with revealing segments of the data that in some way violate our expectations. For an excellent survey of the subject, we refer the reader to the results of Chandola *et al.* (2009). In the realm of anomaly detection within people's activities, the work of Moore and Essa (2001) addresses the problem of error detection and recovery card games that involve two players recorded on video. Their system models the domain with a stochastic context-free grammar and achieves excellent results.

We note that recognizing a failed attempt at an activity is more fine-grained a problem than anomaly detection. The failed event is not just anomalous in general.[5] Rather, it is the specific distinction between success and failure in human activities that

[5] A situation where a player in CTF moves through the campus at a speed of 100 mph and on her way passes an enemy player is certainly anomalous (and probably caused by GPS sensor noise), but we do not want to say that it is a failed attempt at capturing.

we are interested in. And the distinction lies in the fact that an unsuccessful attempt does not yield a certain desired state whereas a successful action does. This desired state is exactly what our approach extracts for each activity in question. To our knowledge, there exists no prior work on explicit modeling and recognition of attempted activities or on learning the intended purpose of an activity in a multi-agent setting.

One of the components of our contribution focuses on *joint* learning and inference across multiple tasks (capturing, freeing, and their respective attempted counterparts). This is in contrast with the traditional "pipeline" learning architecture, where a system is decomposed into a series of modules and each module performs partial computation and passes the result on to the next stage. The main benefits of this set-up are reduced computational complexity and often higher modularity. However, since each stage is myopic, it may not take full advantage of dependencies and broader patterns within the data. Additionally, even though errors introduced by each module may be small, they can accumulate beyond tolerable levels as data passes through the pipeline.

An extensive body of work has shown that joint reasoning improves model performance in a number of natural language processing and data mining tasks including information extraction (*i.e.*, text segmentation coupled with entity resolution) (Poon and Domingos, 2007), co-reference resolution (Poon and Domingos, 2008), information extraction coupled with co-reference resolution (Wellner *et al.*, 2004), temporal relation identification (Yoshikawa *et al.*, 2009; Ling and Weld, 2010), and record de-duplication (Domingos, 2004; Culotta and McCallum, 2005). Similarly to our work, some of the above models are cast in Markov logic. However, prior work uses sampling techniques to perform learning and inference, whereas we apply a reduction to integer linear programming. Interestingly, the work in Denis and Baldridge (2007) jointly addresses the problems of anaphoricity and co-reference via a *manual* formulation of an integer linear program.

Joint activity modeling has also been shown to yield better recognition accuracy, as compared to "pipeline" baselines as well as baselines that make strong inter-activity independence assumptions. The work of Wu *et al.* (2007) performs joint learning and inference over concurrent single-agent activities using a factorial conditional random

field model. Similarly, the work of Helaoui *et al.* (2010) models interleaved activities in Markov logic. They distinguish between foreground and background activities and infer a time window in which each activity takes place from RFID sensory data. By contrast, we focus on joint reasoning about multi-agent activities and attempts in a fully relational—and arguably significantly more noisy—setting.

The work of Manfredotti *et al.* (2010) propose a hierarchical activity recognition system formulated as learning and inference in relational dynamic Bayesian networks. Their model jointly leverages observed interactions with individual objects in the domain and the relationships between objects. Since their method outperforms a hidden Markov model by a significant margin, it contributes additional experimental evidence that a relational decomposition of a domain improves model quality.

The work of Landwehr *et al.* (2007) casts single-agent activity recognition as a relational transformation learning problem, building on transformation-based tagging from natural language processing. Their system induces a set of transformation rules that are then used to infer activities from sensory data. Since the transformation rules are applied adaptively, at each step, the system leverages not only observed data, but also currently assigned labels (inferred activities). However, the transformation rules are learned in a greedy fashion and experiments show that the model does not perform significantly better than a simple HMM. On the other hand, their representation is quite general, intuitive, and extensible. As we will see, our Markov logic model has a similar level of representational convenience while performing global—instead of greedy—optimization in a significantly more complex domain.

The denoising component of our model can be formulated as a tracking problem. Prior work proposed a relational dynamic Bayesian network model for multi-agent tracking (Manfredotti and Messina, 2009). Their evaluation shows that considering relationships between tracked entities significantly improves model performance, as compared to a nonrelational particle filter baseline. By contrast, our work explores *joint* tracking and activity recognition. However, each GPS reading is annotated with the identity of the corresponding agent. The work of Manfredotti and Messina suggests that our model can be generalized, such that the associations between GPS and agent identities

are *inferred* and need not be observed.

Our Markov logic theory can be viewed as a template for a conditional random field (Lafferty, 2001), an undirected graphical model that captures the *conditional* probability of hidden labels given observations, rather than the *joint* probability of both labels and observations, as one would typically do in a directed graphical model. In the relational world, directed formalisms include relational Bayesian networks (Jaeger, 1997) and their dynamic counterparts (Manfredotti, 2009), probabilistic relational models (Koller, 1999; Friedman *et al.*, 1999), Bayesian logic programs (Kersting and De Raedt, 2000), and first-order conditional influence language (Natarajan *et al.*, 2005). Conditional random fields have been extensively applied to activity recognition, and their superior labeling performance over generative models has been demonstrated in a number of both single-agent and multi-agent domains (Liao *et al.*, 2005; Limketkai *et al.*, 2007; Vail, 2008; Vail and Veloso, 2008; Hu *et al.*, 2008). Since MLNs are often solved as propositionalized CRFs, and the directed alternatives can be compiled into a Bayesian network, it can be expected that discriminative relational models generally outperform their generative counterparts on labeling tasks. However, more work needs to be done to answer this question in its entirety.

Since Markov logic is based on, and in fact subsumes, finite first-order logic, we immediately gain access to a number of techniques developed in the rich field of traditional logic. Current Markov logic solvers take advantage of the underlying logical structure to perform more powerful optimizations, such as Alchemy's lifted inference in belief propagation and MC-SAT (Poon and Domingos, 2006). Additionally, domain pruning, where one uses hard constraints to infer reduced domains for predicates, has been shown to lead to significant speed-ups (Papai *et al.*, 2011).

We also leverage this relationship between Markov and first-order logic when inducing an augmented model. Furthermore, presence of dependency cycles introduces additional problems in directed graphical (relational) models. Thus, the fact that, in Markov logic, knowledge can be expressed as weighted first-order formulas combined with the above factors make it a powerful framework best suited for the multi-agent reasoning tasks considered in this work.

Traditional hidden Markov models operate over an alphabet of unstructured (*i.e.*, "flat") symbols. This makes relational reasoning difficult, as one has to either propositionalize the domain, thereby incurring combinatorial increase in the number of symbols and model parameters, or ignore the relational structure and sacrifice information. Logical hidden Markov models (LHMMs) have been proposed to address this problem (Kersting *et al.*, 2006). LHMMs are a generalization of standard HMMs that compactly represents probability distributions over sequences of logical atoms rather than flat symbols. LHMMs have been proven strictly more powerful than their propositional counterparts (HMMs). By applying techniques from logic-based reasoning, such as unification, while leveraging the logical structure component of the model, Kersting *et al.* show that LHMMs often require fewer parameters and achieve higher accuracy than HMMs.

LHMMs have been recently applied to activity recognition. In the context of intelligent user interfaces, the work of Shen (2009) designs and evaluates a LHMM model for recognition of people's activities and workflows carried out on a desktop computer. Other researchers proposed a hierarchical extension of LHMMs along with an efficient particle filter-based inference technique, and apply it to activity recognition problems in synthetic domains (Natarajan *et al.*, 2008). Both lines of work show that LHMMs can be learned and applied efficiently, and perform better than plain HMMs.

However, LHMMs are a generative model and therefore are not ideal for pure labeling and recognition tasks, where we typically do not want to make strong independence assumptions about the observations, nor do we want to explicitly model dependencies in the input space. TildeCRF—a relational extension of traditional conditional random fields—has been introduced to address this issue (Gutmann and Kersting, 2006). TildeCRF allows discriminative learning and inference in CRFs that encode sequences of logical atoms, as opposed to sequences of unstructured symbols. TildeCRF specifically focuses on efficient learning of models of sequential data via boosting, and is subsumed by Markov logic, which can produce both discriminative and generative models. We cast our model in the latter framework to make it more general, extensible, and interpretable.

PRISM, a probabilistic extension of Prolog, has been shown to subsume a wide

variety of generative models, including Bayesian networks, probabilistic context-free grammars, HMMs (along with their logical extension) (Sato and Kameya, 2001, 2008). However, since the focus of PRISM is on representational elegance and generality, rather than scalability, the sheer size of the state space and complexity of our CTF domain precludes its application here.

Finally, our Markov logic theory augmentation process is related to structure learning, transfer learning, and inductive logic programming. In fact, Algorithm 1 implements a special case of structure learning, where we search for a target theory that explains the training data well, while our declarative bias forces the target theory to differ from the source theory only as much as necessary. Again, with the intuition that failed attempts are similar to their failed counterparts. A number of researchers have focused on structure learning specifically in Markov logic networks. This includes early work on top-down structure learning, where clauses in the knowledge base are greedily modified by adding, flipping, and deleting logical literals (Kok and Domingos, 2005). This search is guided by the likelihood of the training data under the current model. The work of Mihalkova and Mooney (2007) exploit patterns in the ground Markov logic networks to introduce a bottom-up declarative bias that makes their algorithm less susceptible to finding only local optima, as compared to alternative greedy methods. Similarly, the work of Kok and Domingos (2009) introduce a bottom-up declarative bias based on lifted hypergraph representation of the relational database. This bias then guides search for clauses that fit the data. Since the hypergraph is lifted, relational path finding tractable. Interesting work on predicate invention applies relational clustering technique formulated in second-order Markov logic to discover new predicates from relational databases (Kok and Domingos, 2007a). The above systems are capable of modeling relatively rich family of logical formulas. Other approaches perform discriminative structure learning and achieve excellent results, but focus on a restricted set of types of formulas (*e.g.*, Horn clauses) (Huynh and Mooney, 2008; Biba *et al.*, 2008). The work of Davis and Domingos (2009) successfully uses second-order Markov logic in deep transfer learning. They lift the model of the source domain to second-order ML and identify high-level structural patterns. These subsequently serve as declarative bias

for structure learning in the target domain.

By its very nature, the inductive logic programming discipline has extensively studied structure learning in deterministic, as well as probabilistic settings (Muggleton, 2002; De Raedt, 2008; De Raedt *et al.*, 2008, *e.g.*,). In fact, our theory augmentation algorithm can be viewed as an efficient Markov logic based version of theory refinement, a well-established ILP technique that aims to improve the quality of a theory in terms of simplicity, fit to newly acquired data, efficiency or other factors (Wrobel, 1996).

Our approach differs from all this work in three main points. First, our declarative bias is defined implicitly by the seed theory of successful activities. Therefore, our theory augmentation algorithm is not limited to any hard-wired set of formula types it can consider. Rather, the search space is defined at run time by extracting motifs from the seed theory. The second distinction lies in computational tractability and exactness of the results. By distinguishing between soft and hard formulas, we are able to search through candidate formulas in a systematic, rather than greedy manner. Consequently, our final learned model requires fewer parameters, which is especially important when the amount of training data is relatively small. Additionally, our weight learning does not experience cold starts, as we leverage the seed theory. The final difference is that, to our knowledge, we are the first to explore structure learning in the context of interplay of success and failure, and their relationship to the intended goals of people's actions.

4.8 Conclusions

This chapter took on the task of understanding the game of capture the flag from GPS data as an exemplar of the general problem of inferring human interactions and intentions from sensor data. We have presented a novel methodology—cast in Markov logic—for effectively combining data denoising with higher-level relational reasoning about a complex multi-agent domain. Specifically, we have demonstrated that given raw and noisy data, we can automatically and reliably detect and recognize both successful and failed interactions in adversarial as well as cooperative settings. Additionally, we have shown that success, failure, and the goal of an activity are intimately tied together

and having a model for successful events allows us to naturally learn models of the other two important aspects of life. Specifically, we have demonstrated that the intentions of rational agents are automatically discovered in the process of resolving inconsistencies between a theory that models successful instances of a set of activities and examples of failed attempts at those activities.

We have formulated four research questions and designed experiments within the CTF domain that empirically answer them. Compared to alternative approaches to solving the multi-agent activity recognition problem, our augmented Markov logic model, which takes into account not only relationships among individual players, but also relationships among activities over the entire length of a game, although computationally more costly, is significantly more accurate on real-world data. Furthermore, we have illustrated that explicitly modeling unsuccessful attempts boosts performance on other important recognition tasks.

4.9 Future Work

Multi-agent activity recognition is especially interesting in the context of current unprecedented growth of on-line social networks—in terms of their size, popularity, and their impact on our "off-line" lives. In this chapter, we show that location information alone allows for rich models of people's interactions, but in the case of on-line social networks, we additionally have access to the content of users' posts and both the explicit and the implicit network interactions. For instance, our recent study shows that, interestingly, about 30% of Twitter status updates reveal their authors' location (Sadilek *et al.*, 2012a). These data sources are now available to machines in massive volumes and at ever-increasing real-time streaming rate. We note that a substantial fraction of posts on services such as Facebook and Twitter talk about everyday activities of the users (Naaman *et al.*, 2010), and this information channel has become available to the research community only very recently. Thus, if we are able to reason about human behavior and interactions in an automated way, we can tap the colossal amounts of knowledge that is—at present—distributed across the whole population.

In Chapter 6, we will see how we can not only reason about explicit GPS traces, but also be able to *infer* and *predict* the location of people who do not broadcast their GPS coordinates. The basic idea is, again, to leverage the structure of relationships among people. The vast majority of us participate in on-line social networks and typically some of our friends there do publish their location. We thus view the GPS-enabled people as noisy location sensors and use the network interactions and dynamics to estimate the location of the rest of the users.

5 Far Out: Long-Term Location Prediction

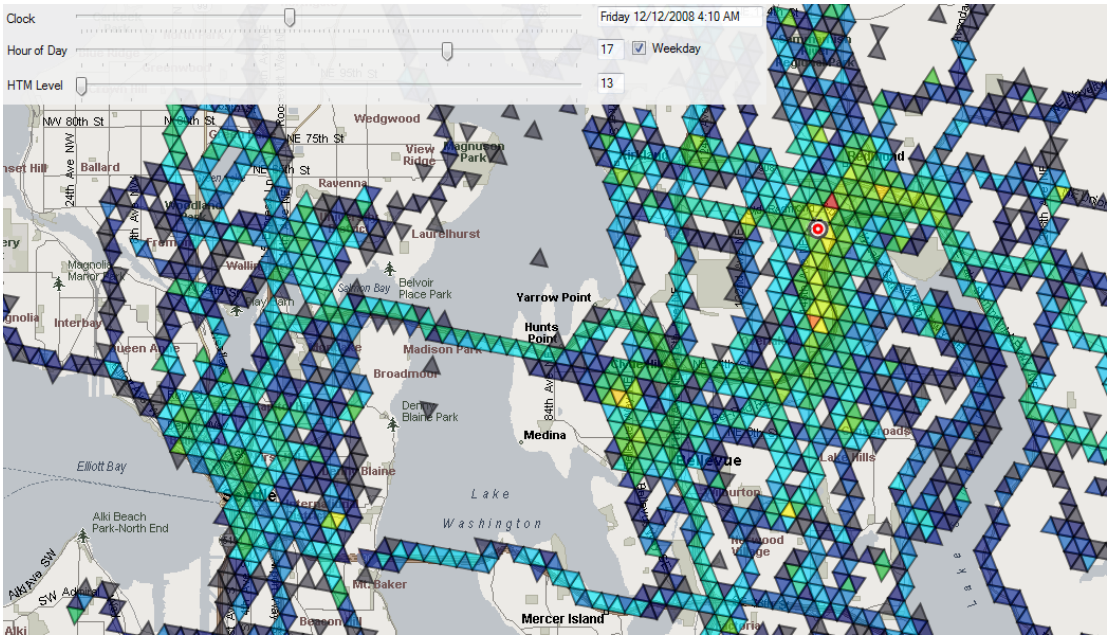


Figure 5.1: This screenshot of our visualization tool shows mobility patterns of one of our subjects living in the Seattle metropolitan area. The colored triangular cells represent a probability distribution of the person's location given an hour of a day and day type.

5.1 Overview

Much work has been done on predicting where is one going to be in the immediate future, typically within the next hour. By contrast, we address the open problem of predicting human mobility far into the future, a scale of months and years. We propose an efficient nonparametric method that extracts significant and robust patterns in location data, learns their associations with contextual features (such as day of week), and subsequently leverages this information to predict the most likely location at any given time in the future. The entire process is formulated in a principled way as an eigendecomposition problem. Evaluation on a massive dataset with more than 32,000 days worth of GPS data across 703 diverse subjects shows that our model predicts the correct location with high accuracy, even years into the future. This result opens a number of interesting avenues for future research and applications.

5.2 Motivation

Where are you going to be 285 days from now at 2PM? This work explores how accurately such questions can be answered across a large sample of people. We propose a novel model of long-term human mobility that extracts significant patterns shaping people's lives, and helps us understand large amounts of data by visualizing the patterns in a meaningful way. But perhaps most importantly, we show that our system, Far Out, *predicts* people's location with high accuracy, even *far into the future*, up to multiple years.

Such predictions have a number of interesting applications at various scales of the target population size. We will give a few examples here. Focusing on one individual at a time, we can provide better reminders, search results, and advertisements by considering all the locations the person is likely to be close to in the future (*e.g.*, "Need a haircut? In 4 days, you will be within 100 meters of a salon that will have a $5 special at that time."). At the social scale (people you know), we can leverage Far Out's predictions to suggest a convenient place and time for everybody to meet, even when they are dispersed

throughout the world. We also envision a peer-to-peer package delivery system, but there one would heavily rely on a reliable set of exchange locations, where people are likely to meet in the future. Far Out can provide these. Finally, at the population scale, Far Out is the first step towards bottom-up modeling of the evolution of an entire metropolitan area. By modeling long-term mobility of individuals, emergent patterns, such as traffic congestion, spread of disease, and demand for electricity or other resources, can be predicted a long time ahead as well. These applications motivate the predictive aspect of Far Out, but as we will see, the patterns it finds are also useful for gaining insight into people's activities and detecting unusual behavior. Researchers have recently argued for a comprehensive scientific approach to urban planning, and long-term modeling and prediction of human mobility is certainly an essential element of such a paradigm (Bettencourt and West, 2010).

Techniques that work quite well for short-term prediction, such as Markov models and random walk-based models, are of little help for long-term inference. Both classes of models make strong independence assumptions about the domain, and one often postulates that a person's location at time t only depends on her location at time $t-1$. Such models give increasingly poorer and poorer predictions as they are forced to evolve the system further into the future (Musolesi and Mascolo, 2009). Although one can improve the performance by conditioning on a larger context and structure the models hierarchically, learning and inference quickly become intractable or even infeasible due to computational challenges and lack of training data (Liao *et al.*, 2007b,a).

While your recent location is in general highly independent of your location in the distant future, as we will see, it is likely to be a good predictor of your location exactly one week from now. Therefore, we view long-term prediction as a process that identifies strong motifs and regularities in subjects' historical data, models their evolution over time, and estimates future locations by projecting the patterns into the future. Far Out implements all three stages of this process.

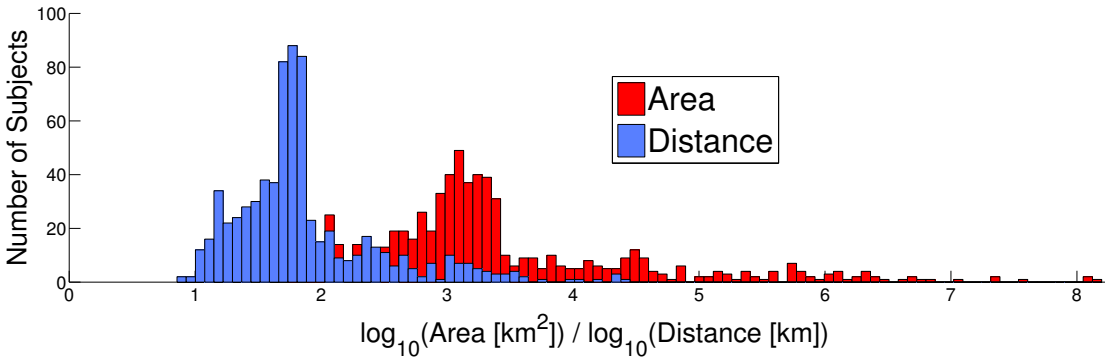


Figure 5.2: The distribution of the bounding rectangular geographical areas and longest geodesic distances covered by individual subjects.

5.3 The Data

We evaluate our models on a large dataset consisting of 703 subjects of two types: people ($n = 307$) and vehicles ($n = 396$). The people include paid and unpaid volunteers who carried consumer-grade GPS loggers while going about their daily lives. Vehicles consist of commercial shuttles, paratransit vans, and personal vehicles of our volunteers, and had the same GPS unit installed on their dashboard. While some of the shuttles follow a relatively fixed schedule, most of them are available on demand and, along with the paratransit vans, flexibly cover the entire Seattle metropolitan area.

Since this work focuses on long-term prediction, we need to consider only datasets that span extensive time periods, which are rare. The number of contiguous days available to us varies across subjects from 7 to 1247 ($\mu = 45.9$, $\sigma = 117.8$). Overall, our dataset contains 32,268 days worth of location data. Fig. 5.2 shows the distribution of the area (bounding rectangle) and distances covered by our subjects. We observe high variance in the area across subjects, ranging from 30 to more than 10^8 km^2. To put these numbers in perspective, the surface area of the entire earth is 5.2×10^8 km^2.

5.4 Methodology and Models

Our models leverage Fourier analysis to find significant periodicities in human mobility, and principal component analysis (PCA) to extract strong meaningful patterns from location data, which are subsequently leveraged for prediction.

To enable Fourier analysis, we represent each GPS reading, consisting of a latitude, longitude pair for each time t, as a complex number $z_t = \text{latitude}_t + (\text{longitude}_t)i$. This allows us to perform Fourier analysis *jointly* over both spatial dimensions of the data, thereby extracting significant periods in a principled way (see Fig. 5.6).

We can transform a function f from time domain to frequency domain using Fourier transform, and subsequently formulate f as a weighted sum of harmonic basis functions. In this work, we deal with GPS coordinates uniformly sampled in time, and therefore apply the discrete Fourier transform (DFT) given by

$$Z_k = \mathscr{F}_t\left[\{z_t\}_{t=0}^{T-1}\right](k) = \sum_{t=0}^{T-1} z_t e^{\left(-2\pi k \frac{t}{T}\right)i} \tag{5.1}$$

where $z_0, \ldots z_{T-1}$ is a sequence of complex numbers representing a subject's location over T seconds. We refer the reader to (Brigham and Morrow, 1967) for more details on DFT.

PCA is a dimensionality reduction technique that transforms the original data into a new basis, where the basis vectors are, in turn, aligned with the directions of the highest remaining variance of the data. PCA can be performed by eigendecomposition of the data covariance matrix, or by applying singular value decomposition (SVD) directly on the data matrix. Our implementation uses the latter approach, as it's more numerically stable. PCA has a probabilistic interpretation as a latent variable model, which endows our model with all the practical advantages stemming from this relationship, such as efficient learning and dealing with missing data (Tipping and Bishop, 1999). For a thorough treatment of PCA, see (Jolliffe, 2002).

We consider continuous (GPS coordinates) as well as discrete (occupancy grid) data, and our models work with both modalities without *any* modification to the mathematics or the algorithms. In both cases we represent each day as a vector of features. In the

continuous representation, we have a 56-element vector shown in Fig. 5.3. The first 24 elements capture the subject's median latitude for each hour of the day, the next 24 elements correspond to the median longitude, the following 7 elements encode the day of week (in 1-out-of-7 binary code, since it's a categorical variable), and the final element is 1 if the day is a national holiday in the subject's current locale (*e.g.*, Christmas, Thanksgiving) and 0 otherwise. This representation helps us capture the dependence between the subject's location and the hour of the day, day of week, and whether or not the day is a holiday. The continuous representation is best suited for predicting a subject's single, approximate location for a given time, possibly for finding nearby people or points of interest. This representation is not probabilistic, as the discretized representation we describe next.

In the discretized condition, we divide the surface of the globe into equilateral triangular cells of uniform size (side length of 400 meters), and assign each GPS reading to the nearest cell. We then induce an empirical probability distribution over the ten most frequently visited cells and one "other" location that absorbs all GPS readings outside of the top ten cells. Our analysis shows that the 10 discrete locations capture the vast majority of an individual's mobility, and each such cell can often be semantically labeled as home, work, favorite restaurant, *etc.*

Fig. 5.1 shows the occupancy probability distribution over the cells for one of our subjects, given by

$$\Pr(C = c \mid T = t, W = w) = \frac{\text{count}(c, t, w)}{\sum_{c' \in \Omega_C} \text{count}(c', t, w)} \tag{5.2}$$

where C, T, and W are random variables representing cells, time of day, and day type, respectively. Ω_C is the set of all cells.

We construct a feature vector for each day from this probability distribution as shown in Fig. 5.4, where the first 11 elements model the occupancy probability for the 11 discrete places between 00:00 and 00:59 of the day, the next 11 elements capture 01:00 through 01:59, *etc.* The final 8 elements are identical to those in the continuous representation. The discretized representation sacrifices the potential precision of the continuous representation for a richer representation of uncertainty. It does not con-

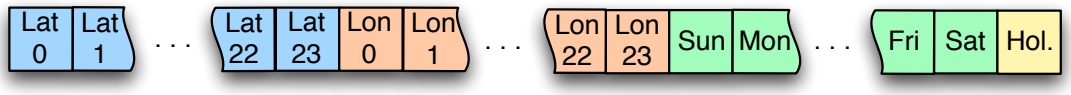


Figure 5.3: Our continuous vector representation of a day $\boldsymbol{d}$ consists of the median latitude and longitude for each hour of the day (00:00 through 23:59), binary encoding of the day of week, and a binary feature signifying whether a national holiday falls on $\boldsymbol{d}$.

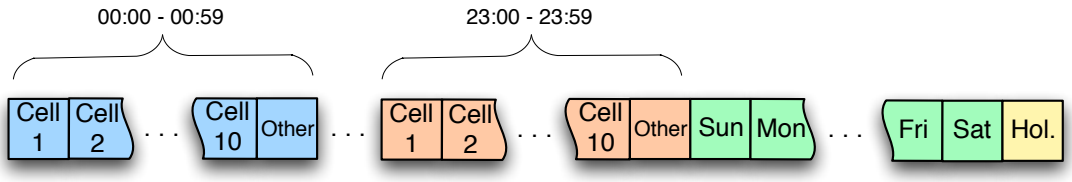


Figure 5.4: Our cell-based vector representation of a day $\boldsymbol{d}$ encodes the probability distribution over dominant cells conditioned on the time within $\boldsymbol{d}$, and the same day-of-week and holiday information as the continuous representation (last 8 elements).

strain the subject's location to a single location or cell, but instead represents the fact that the subject could be in one of several cells with some uncertainty for each one.

The decision to divide the data into 24-hour segments is not arbitrary. While a day is a natural measure of time, perhaps it is not the most appropriate time period to model. Applying Fourier analysis to the raw GPS data as described above yields a power spectrum, an example of which we show in Fig. 5.6. We see that most of the powerful periods are shorter or equal to 24 hours, and the longer ones (*e.g.*, 42, 48) tend to be integer multiples of the shorter periods. Therefore, day-long segments are convenient both conceptually and numerically.

The decision to divide the data into 24-hour segments is not arbitrary. Applying DFT to the raw GPS data as described above shows that most of the energy is concentrated in periods shorter or equal to 24 hours.

Now we turn our attention to the eigenanalysis of the subjects' location, which provides further insights into the data. Each subject is represented by a matrix $\boldsymbol{D}$,

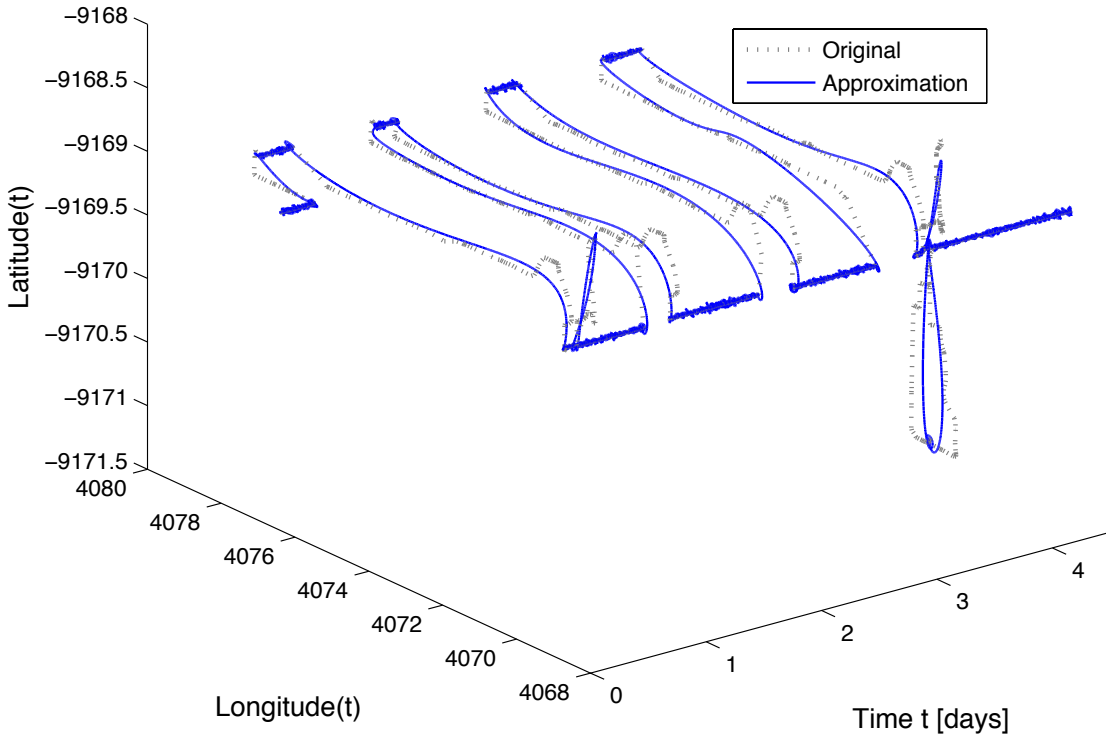


Figure 5.5: Visualization of GPS data as a dynamic sequence of complex numbers, where the real part of each number represents latitude and the imaginary part longitude. The grey line shows the original data, the blue line its approximation using only the top thousand most powerful frequencies. We see that much effort is spent on modeling the constant segments of the trajectories (*e.g.*, when a person is sleeping at home).

where each row is a day (either in the continuous or the cell form). Prior to computing PCA, we apply Mercator cylindrical projection on the GPS data and normalize each column of observations by subtracting out its mean $\boldsymbol{\mu}$ and dividing by its standard deviation $\boldsymbol{\sigma}$. Normalizing with the mean and standard deviation scales the data so values in each column are in approximately the same range, which in turn prevents any columns from dominating the principal components.

Applying SVD, we effectively find a set of eigenvectors of $\boldsymbol{D}$'s covariance matrix, which we call *eigendays* (Fig. 5.7). As we see in Fig. 5.8, a few top eigendays with the largest eigenvalues induce a subspace, onto which a day can be projected, and that captures most of the variance in the data. For virtually all subjects, ten eigendays

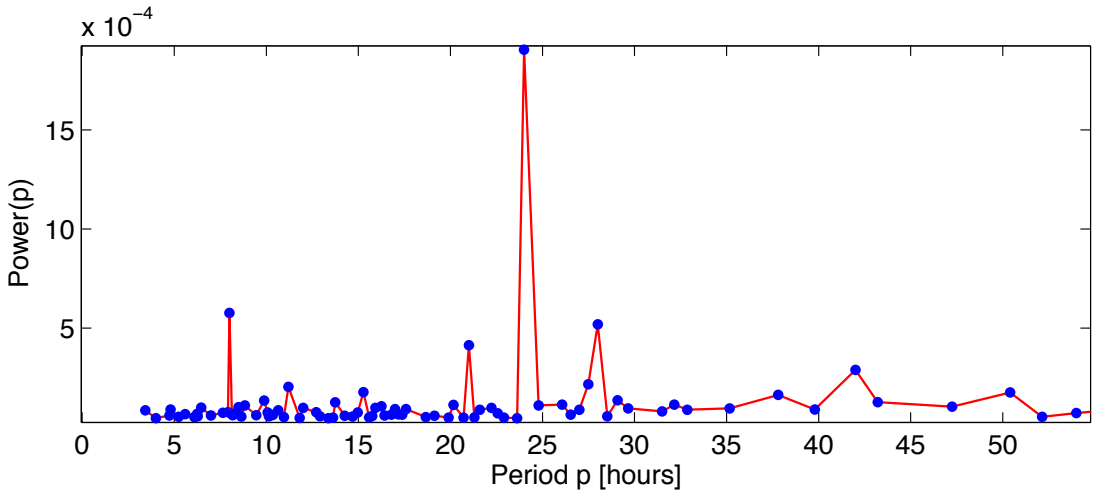


Figure 5.6: A typical power spectrum of an individual obtained by Fourier analysis of his GPS data represented on a complex plane. Note that the two most prominent periods are 8 and 24 hours (exactly). The relatively strong periods of 21 and 28 hours show that this person probably exhibits multiple modes of behavior, switching between "short" and "long" days as time goes on.

are enough to reconstruct their entire location log with more than 90% accuracy. In other words, we can accurately compress an arbitrary day $\boldsymbol{d}$ into only $n \ll |\boldsymbol{d}|$ weights $w_1, \ldots, w_n$ that induce a weighted sum over a common set of ten most dominant eigendays $\boldsymbol{\mathcal{E}}_i$:

$$\boldsymbol{d} \cong \left[\left(\sum_{i=1}^{n} w_i \boldsymbol{\mathcal{E}}_i \right) + \boldsymbol{\mu} \right] \mathrm{diag}(\boldsymbol{\sigma}). \tag{5.3}$$

This applies to both continuous and discretized data. The reason for this is that human mobility is relatively regular, and there is a large amount of redundancy in the raw representation of people's location. Note that unlike most other approaches, such as Markov models, PCA captures long-term correlations in the data. In our case, this means patterns in location over an entire day, as well as joint correlations among additional attributes (day of week, holiday) and the locations.

Our eigenanalysis shows that there are strong correlations among a subject's latitudes and longitudes over time, and also correlations between other features, such as the

day-of-week, and raw location. Let's take eigenday #2 ($\boldsymbol{\mathcal{E}}_2$) in Fig. 5.7 as an example. From the last 8 elements, we see that PCA automatically grouped holidays, weekends, and Tuesdays within this eigenday. The location pattern for days that fit these criteria is shown in the first 48 elements. In particular, $\boldsymbol{\mathcal{E}}_2$ makes it evident that this person spends her evenings and nights (from 16:00 to 24:00) at a particular constant location in the North-West "corner" of her data, which turns out to be her home.

The last 8 elements of each eigenday can be viewed as indicators that show how strongly the location patterns in the rest of the corresponding eigenday exhibit themselves on a given day-of-week $\times$ holiday combination. For instance, $\boldsymbol{\mathcal{E}}_3$ is dominant on Saturdays, $\boldsymbol{\mathcal{E}}_7$ on Fridays, and $\boldsymbol{\mathcal{E}}_{10}$ on Tuesdays that are not holidays (compare with $\boldsymbol{\mathcal{E}}_2$).

Fig. 5.9 shows the top ten eigendays for the cell-based representation. Now we see patterns in terms of probability distributions over significant cells. For instance, this subject exhibits a strong "baseline" behavior ($\boldsymbol{\mathcal{E}}_1$) on all days—and especially non-working days—except for Tuesdays, which are captured in $\boldsymbol{\mathcal{E}}_2$. Note that the complex patterns in cell occupancy as well as the associated day types can be directly read off the eigendays.

Our eigenday decomposition is also useful for detection of anomalous behavior. Given a set of eigendays and their typical weights computed from training data, we can compute how much a new day deviates from the subspace formed by the historical eigendays. The larger the deviation, the more atypical the day is. We leave this opportunity for future work.

So far we have been focusing on the descriptive aspect of our models—what types of patterns they extract and how can we interpret them. Now we turn to the predictive power of Far Out.

5.4.1 Predictive Models

We consider three general types of models for long-term location prediction. Each type works with both continuous (raw GPS) as well as discretized (triangular cells) data, and

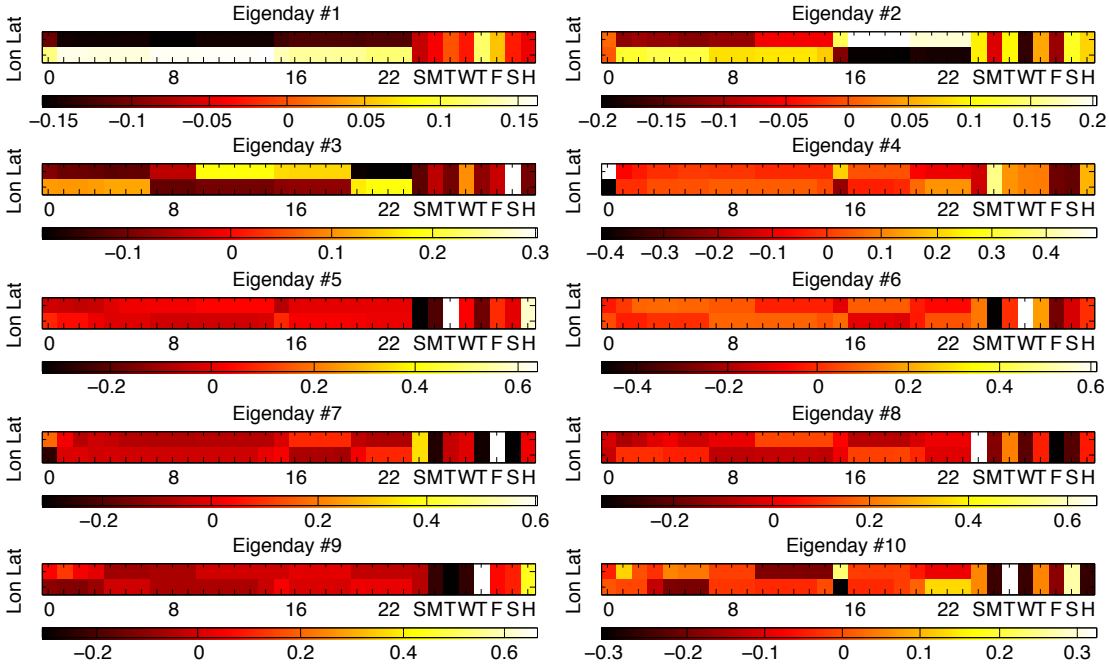


Figure 5.7: Visualization of the top ten most dominant eigendays ($\boldsymbol{\mathcal{E}}_1$ through $\boldsymbol{\mathcal{E}}_{10}$). The leftmost 48 elements of each eigenday correspond to the latitude and longitude over the 24 hours of a day, latitude plotted in the top rows, longitude in the bottom. The next 7 binary slots capture the seven days of a week, and the last element models holidays versus regular days (*cf.* Fig. 5.3). The patterns in the GPS as well as the calendar features are color-coded using the mapping shown below each eigenday.

all our models are directly applied to both types of data without *any* modification of the learning process. Furthermore, while we experiment with two observed features (day of week and holiday), our models can handle arbitrary number of additional features, such as season, predicted weather, social and political features, known traffic conditions, information extracted from the power spectrum of an individual, and other calendar features (*e.g.*, *Is this the second Thursday of a month?*; *Does a concert or a conference take place?*). In the course of eigendecomposition, Far Out automatically eliminates insignificant and redundant features.

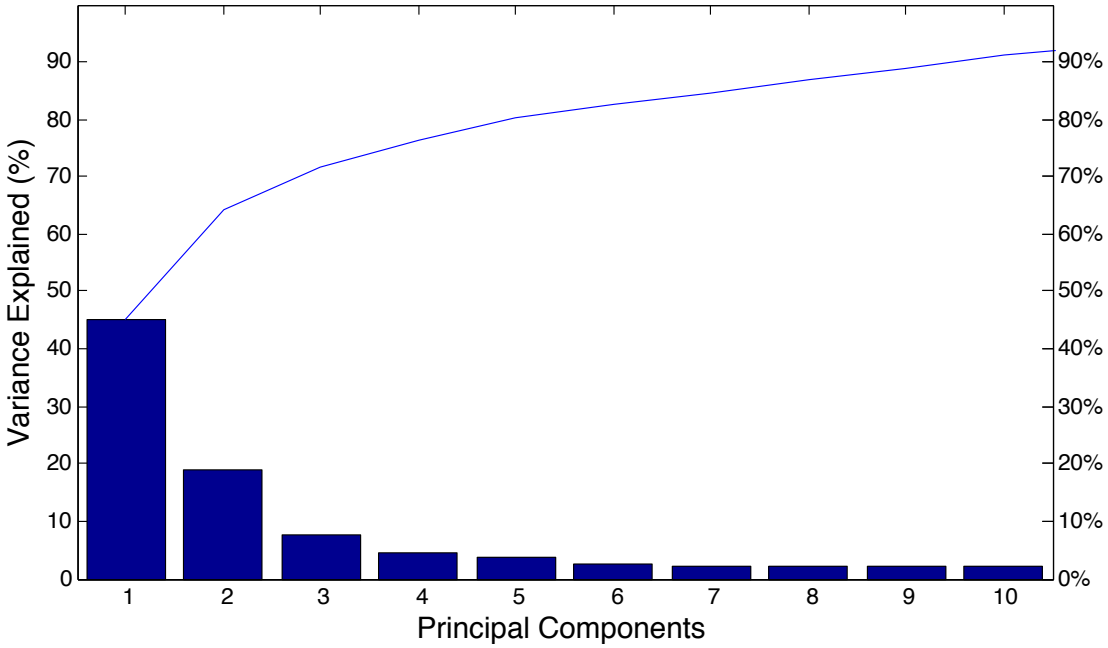


Figure 5.8: This Pareto plot shows a pattern in the analysis of variance typical for all subjects in our dataset: a handful of eigendays account for vast majority of the total variance in the data.

Mean Day Baseline Model

For the continuous GPS representation, the baseline model calculates the average latitude and longitude for each hour of day for each day type. In the discrete case, we use the mode of cell IDs instead of the average. To make a prediction for a query with certain observed features o, this model simply retrieves all days that match o from the training data, and outputs their mean or mode. Although simple, this baseline is quite powerful, especially on large datasets such as ours. It virtually eliminates all random noise for repeatedly visited places. Additionally, since the spatial distribution of sporadic and unpredictable trips is largely symmetric over long periods of time, the errors these trips would have caused tend to be averaged out by this model (*e.g.*, a spontaneous trip Seattle-Las Vegas is balanced by an isolated flight Seattle-Alaska).

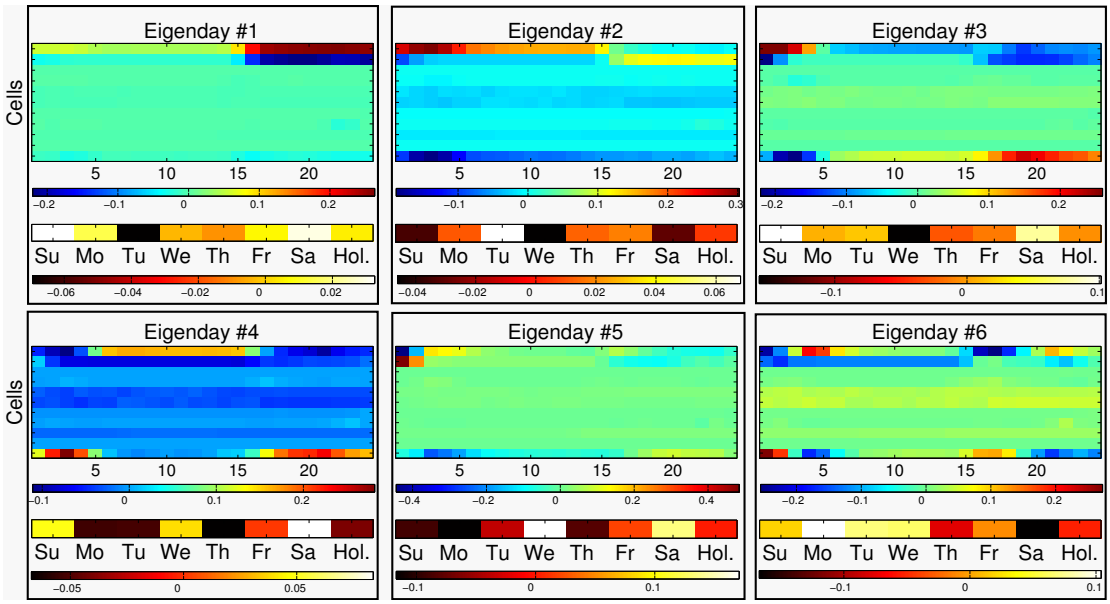


Figure 5.9: Visualization of the top six most dominant eigendays ($\boldsymbol{\mathcal{E}}_1$ through $\boldsymbol{\mathcal{E}}_6$). The larger matrix within an eigenday shows cell occupancy patterns over the 24 hours of a day. Patterns in the calendar segment of each eigenday are shown below each matrix (*cf.* Fig. 5.4).

Projected Eigendays Model

First, we learn all principal components (a.k.a. eigendays) from the training data as described above. This results in a $n \times n$ matrix $\boldsymbol{P}$, with eigendays as columns, where n is the dimensionality of the original representation of each day (either 56 or 272).

At testing time, we want to find a fitting vector of weights $\boldsymbol{w}$, such that the observed part of the query can be represented as a weighted sum of the corresponding elements of the principal components in matrix $\boldsymbol{P}$. More specifically, this model predicts a subject's location at a particular time t_q in the future by the following process. First, we extract observed features from t_q, such as which day of week t_q corresponds to.The observed feature values are then written into a query vector $\boldsymbol{q}$. Now we *project* $\boldsymbol{q}$ onto the eigenday space using only the observed elements of the eigendays. This yields a weight for each eigenday, that captures how dominant that eigenday is given the observed feature values:

$$\boldsymbol{w} = (\boldsymbol{q} - \boldsymbol{\mu})\,\mathrm{diag}(\boldsymbol{\sigma}^{-1})\boldsymbol{P}_c \tag{5.4}$$

where $\boldsymbol{q}$ is a row vector of length m (the number of observed elements in the query vector), $\boldsymbol{P}_c$ is a $m \times c$ matrix (c is the number of principal components considered), and $\boldsymbol{w}$ is a row vector of length c. Since we implement PCA in the space of normalized variables, we need to normalize the query vector as well. This is achieved by subtracting the mean $\boldsymbol{\mu}$, and component-wise division by the variance of each column $\boldsymbol{\sigma}$.

Note that finding an optimal set of weights can be viewed as solving (for $\boldsymbol{w}$) a system of linear equations given by

$$\boldsymbol{w}\boldsymbol{P}_c^T = (\boldsymbol{q} - \boldsymbol{\mu})\,\mathrm{diag}(\boldsymbol{\sigma}^{-1}). \tag{5.5}$$

However, under most circumstances, such a system is ill-conditioned, which leads to an undesirable numerical sensitivity and subsequently poor results. The system is either over- or under-determined, except when $c = m$. Furthermore, $\boldsymbol{P}_c^T$ may be singular.

Theorem 1. *The projected eigendays model learns weights by performing a least-squares fit.*

Proof. If $\boldsymbol{P}$ has linearly independent rows, a generalized inverse (*e.g.*, Moore-Penrose) is given by $\boldsymbol{P}^+ = \boldsymbol{P}^*(\boldsymbol{P}\boldsymbol{P}^*)^{-1}$ (Ben-Israel and Greville, 2003). In our case, $\boldsymbol{P} \in \mathbb{R}^{m \times c}$ and by definition forms an orthonormal basis. Therefore $\boldsymbol{P}\boldsymbol{P}^*$ is an identity matrix and it follows that $\boldsymbol{P}^+ = \boldsymbol{P}^T$. It is known that pseudoinverse provides a least-squares solution to a system of linear equations (Penrose, 1956). Thus, equations 5.4 and 5.5 are theoretically equivalent, but the earlier formulation is significantly more elegant, efficient, and numerically stable. □

Using Eq. 5.3, the inferred weights are subsequently used to generate the prediction (either continuous GPS or probability distribution over cells) for time t_q. Note that both training and testing are efficient ($\mathcal{O}(cdm)$, where d is the number of days) and completely nonparametric, which makes Far Out very easy to apply to other domains with different features.

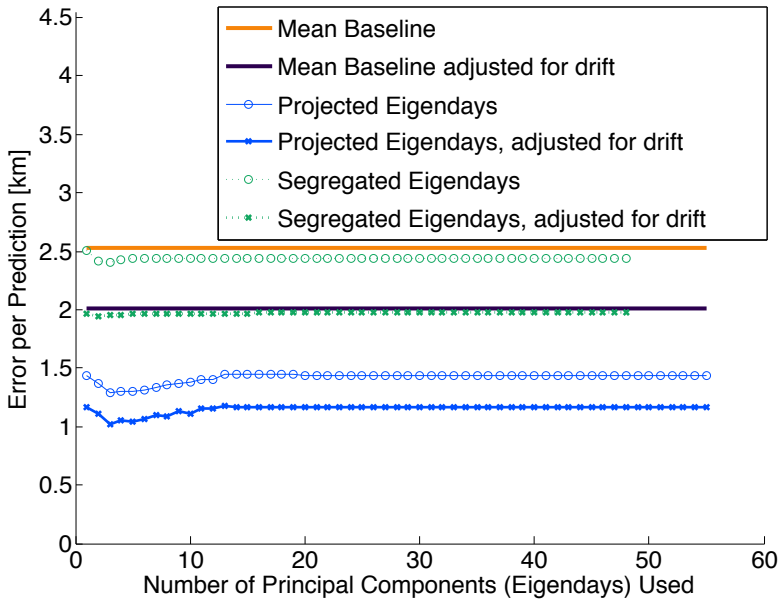


Figure 5.10: Comparison in terms of absolute prediction error over all subjects as we vary the number of eigendays we leverage.

Segregated Eigendays Model

While the last two models induced a single set of eigendays, this model learns a separate library of eigendays for each day type, *e.g.*, eigen-holiday-mondays, over only the location elements of the day vectors $\boldsymbol{d}$. Prediction is made using Eq. 5.3, where the weights are proportional to the variance each eigenday explains in the training data.

5.4.2 Adapting to Pattern Drift

Since our models operate in a space of normalized variables, we can adapt to the drift of mean and variance of each subject's locations, which does occur over extended periods of time. The basic idea is to weigh more recent training data more heavily than older ones when de-normalizing a prediction (see Eq. 5.3). We achieve this by imposing a linear decay when learning $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ from the training data.

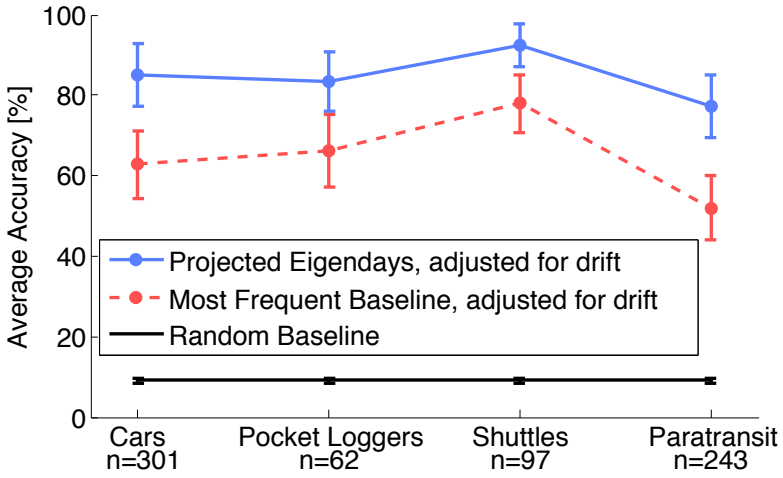


Figure 5.11: Accuracy of cell-based predictions varies across subject types, but the projected eigendays model outperforms its alternatives by a significant margin.

5.5 Experiments and Results

In this section, we evaluate our approach, compare the performance of the proposed models, and discuss insights gained. Unless noted otherwise, for each subject, we always train on the first half of her data (chronologically) and test on the remaining half.

First, let's look at the predictions in the continuous GPS form, where the crucial metric is the median absolute error in distance. Fig. 5.10 shows the error averaged over all subjects as a function of the number of eigendays leveraged. We show our three model types, both with and without addressing pattern drift. We see that the segregated eigendays model is not significantly better than the baseline. One reason is that it considers each day type in isolation and therefore cannot capture complex motifs spanning multiple days. Additionally, it has to estimate a larger number of parameters than the unified models, which negatively impacts its performance, especially for subjects with smaller amounts of training data.

By considering only the strongest eigendays, we extract the dominant and, in a sense, most dependable patterns, and filter out the volatile, random, and less significant signals. This effect is especially strong in the projected model. Finally, we see that modeling pattern drift systematically reduces the error by approximately 27%.

Now we focus on the evaluation of the same models, but this time they operate on the cell representation. We additionally consider a trivial random baseline that guesses the possible discrete locations uniformly at random. Our eigenday-based models predict based on maximum likelihood:

$$c^{\star}_{t,w} = \operatorname*{argmax}_{c} \big(\Pr(C = c \mid T = t, W = w) \big).$$

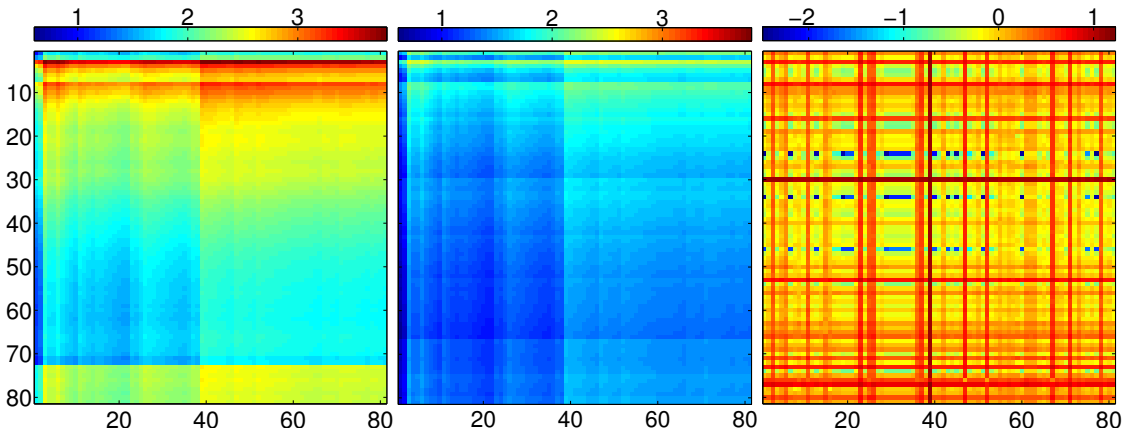
For the sake of brevity, we will focus on the projected eigendays model adapted to pattern drift (with results averaged over c, the number of eigendays used), as our evaluation on the cell-based representation yields the same ordering in model quality as in Fig. 5.10.

In Fig. 5.11, we see that the eigenday model clearly dominates both baselines, achieving up to 93% accuracy. Personal cars are about as predictable as pocket loggers (84%), and paratransit vans are significantly harder (77%), as they don't have any fixed schedule nor circadian rhythms.

Since we evaluate on a dataset that encompasses long periods of time, we have a unique opportunity to explore how the test error varies as we make predictions progressively further into the future and increase the amount of training data. Fig. 5.12 shows these complex relationships for one of our subjects with a total of 162 weeks of recorded data. By contrast, virtually all work to date has concentrated on the first column of *pixels* on the left-hand side of the plot. This is the region of short-term predictions, hours or days into the future.

We see that the projected eigenday model systematically outperforms the baseline and produces a low test error for predictions spanning the entire 81 week testing period (*cf.* Figs. 5.12a and 5.12b). In general, as we increase the amount of training data, the error decreases, especially for extremely long-term predictions.

Fig. 5.12c shows that not all weeks are created equal. There are several unusual and therefore difficult weeks (*e.g.*, test week #38), but in general our approach achieves high accuracy even for predictions 80 weeks into the future. Subsequent work can take advantage of the hindsight afforded by Fig. 5.12, and eliminate anomalous or confusing time periods (*e.g.*, week #30) from the training set.



(a) Mean Baseline (b) PCA Cumulative (c) PCA Separate

Figure 5.12: How test error varies depending on how far into the future we predict and how much training data we use. Each plot shows the prediction error, in km, as a function of the amount of training data in weeks (vertical axes), and how many weeks into the future the models predict (horizontal axes). Plots (a) and (b) visualize cumulative error, where a pixel with coordinates (x, y) represents the average error over testing weeks 1 through x, when learning on training weeks 1 through y. Plot (c) shows, on a log scale, the error for each pair of weeks separately, where we train only on week y and test on x.

Finally, decomposition of the prediction error along day types shows that for human subjects, weekends are most difficult to predict, whereas work days are least entropic. While this is to be expected, we notice a more interesting pattern, where the further away a day is from a nonworking day, the more predictable it is. For instance, Wednesdays in a regular week are the easiest, Fridays and Mondays are harder, and weekends are most difficult. This motif is evident across all human subjects and across a number of metrics, including location entropy, KL divergence and accuracy (cell-based representation), as well as absolute error (continuous data). Shuttles and paratransit exhibit the exact inverse of this pattern.

5.6 Related Work

There is ample previous work on building models of short-term mobility, both individual and aggregate, descriptive as well as predictive. But there is a gap in modeling and predicting long-term mobility, which is our contribution (see Table 5.1).

	Short Term	**Long Term**
Descriptive	Previous work	Previous work
Predictive	Previous work	**Only Far Out**
Unified	Previous work	**Only Far Out**

Table 5.1: The context of our contributions.

Recent research in location-based reasoning has shown that surprisingly rich models of human behavior can be learned from GPS data alone, for example (Ashbrook and Starner, 2003; Liao *et al.*, 2005; Krumm and Horvitz, 2006; Eagle and Pentland, 2006; Horvitz *et al.*, 2005; Sadilek and Kautz, 2010b). However, previous work largely focused on making predictions at fixed, and relatively short, time scales. Consequently, questions such as "where is Karl going to be in the next hour?" can often be answered with high accuracy. By contrast, this work explores the predictability of people's mobility at varying temporal resolutions, and far into the future. While short-term prediction

is often sufficient for routing in wireless networks, one of the major applications of location modeling, long-term modeling is crucial in ubiquitous computing, infrastructure planning, traffic prediction, and in both personal and digital security.

The study of high-level statistical properties of human mobility has received considerable attention in recent years (Kim *et al.*, 2006; Hsu *et al.*, 2007b; González *et al.*, 2008; Lee *et al.*, 2009; Musolesi and Mascolo, 2009) present a number of generative models of people's location that are capable of synthesizing GPS traces that accurately mimic observed statistics of real GPS trajectories. Subsequently, (Rhee *et al.*, 2011) show that human walking behavior is statistically similar to Levy walks, characterized by its heavy-tailed (power law) distribution of lengths of the individual walk segments. The main motivation for this line of work is to improve performance of wireless networks.

Interestingly, (Jiang *et al.*, 2009) argue that human mobility can be, at least at an aggregate level, modeled even when no knowledge about people's goals is given. They show that Levy flight characteristics emerge simply from randomized navigation within typical road networks.

(Jardosh *et al.*, 2003) add the notion of obstacles (*e.g.*, buildings) that limit both mobility and signal strength. Walkways are synthesized by a Voronoi diagram over user-defined obstacles. Jardosh et al. show that the presence of obstacles makes a significant difference in the efficiency of wireless routing protocols.

(Ekman *et al.*, 2008) suggest a modular framework of human mobility, where various sub-models are invoked at different times of day (*e.g.*, "at home" sub-model dominates at night) to generate user movement that is statistically similar to real location traces.

By analyzing connections to Wi-Fi access points on a university campus, (Kim and Kotz, 2007) extract different *types* of people's mobility using unsupervised clustering. To remove the dependency of location on absolute time, they transform the data into frequency domain using discrete Fourier transform. Similarly, (Kim and Kotz, 2011) perform dynamic moving-average profiling to induce typical mobility patterns of users of a large Wi-Fi network. Wireless access points are used as sensors of user location, and Pearson coefficient governs if a new trace matches an existing profile, or should

form a basis for a novel pattern. Applying matrix decomposition techniques, (Hsu *et al.*, 2007a) also extract significant mobility patterns (in this case, eigen-behaviors) from Wi-Fi connection logs. (Moon and Helmy, 2010) study the patterns of encounters of mobile devices. Applying discrete Fourier transform, they categorize pairs of devices based on their periodicity of encounters. (Bar-David *et al.*, 2009) apply Fourier analysis to study movement of herds of Buffalo. (Noulas *et al.*, 2011) study patterns of human mobility within a large Foursquare dataset spanning diverse areas of the globe. They show that the probability of transition between two places bears an inverse power relation to the number of competing opportunities between them to their plain geographical distance. This metric was introduced by (Stouffer, 1940). They subsequently leverage this property to generate realistic synthetic movement of people in urban areas. However, all works above do not address the problem of location *prediction*, which is the primary focus of this chapter.

(Scellato *et al.*, 2011a) concentrate on predicting a person's next significant location while explicitly modeling the time of arrival along with the duration of stay. Similarly to (Scott *et al.*, 2011), the prediction is based on finding the best match for partial sequence of observations within logs of historical data. However, both works address only short-term predictions. On the other hand, (Krumm and Brush, 2011) model probability of being at home at any given hour of a day, thereby allowing longer-term predictions by leveraging periodic patterns in human mobility. We, in contrast, focus on capturing both periodic and irregular behavior including long-term complex correlations and patterns in the data that encompasses a large number of places, not just one's home.

A number of researchers characterized the location prediction problem as various forms of random walk. (Fulop *et al.*, 2007) compare the traditional Brownian motion-type model to a lifted Markov random walk, where a particle samples from cells to its left, right, or remains stationary. The sampling parameters are learned from historical trajectories. Their results show that in the context of cell phone networks the Markovian representation achieves significantly higher accuracy. Similarly, (Chiang and Shenoy, 2004) represent a random walk in two dimensions as a Markov model and focus on minimizing the number of states by coalescing equivalent states. Additionally, they

model dwell time in the Markovian setting by an absorbing state. In general, studies of people as well as animals demonstrate that vanilla random walk models achieve respectable accuracy for short-term prediction, they dramatically diverge as we increase the time offset (Bergman *et al.*, 2000; Fulop *et al.*, 2007; Musolesi and Mascolo, 2009).

Assuming people drive the posted speed limit, (Liu and Karimi, 2006) approximate the area in which a person can appear within the next time step and predict the person's location within it. The GPS trace leading to each intersection is represented by features—such as its duration and number of turns—which are subsequently leveraged in a logistic regression model to predict the direction of the next turn. Liu et al. show that this approach is significantly better than having an explicit conditional probability table for each intersection. Future work may generalize the uniform circle of reachable location used by Liu et al. into a *maximal mobility cone*, where candidate future locations are strictly constrained by one's current position along with means of transportation available (*e.g.*, bike, car, airplane). This builds on the notion of light cone from special relativity, where due to finite speed of light, one can only affect certain regions of space-time.

(Liu *et al.*, 1998) propose a two-level hierarchical model of location of wireless network users. The top level captures people's regular behavior at the level of fairly large cells, while the low level models movement within a cell as a stochastic process based on Kalman filter.

While not focusing on location prediction *per se*, (Jiang *et al.*, 2001) address a related problem of link availability in wireless networks. Experiments on synthetic data show this to be a very challenging problem.

(Li *et al.*, 2010) propose a model for extracting significant and meaningful periodic mobility patterns of individuals. They first discretize the raw location data by finding frequently visited places. Then each GPS trace is represented as n 0-1 functions, where n is the number of discrete locations. Li et al. subsequently apply Fourier transform coupled with auto-correlation to extract dominant frequency components and the corresponding time periods. Finally, they cluster periodic behaviors (*e.g.*, "vacation days" versus "work days") and learn a probabilistic representation of activities in each cluster.

(Vlachos *et al.*, 2005) synergetically combine DFT with autocorrelation to detect periodic patterns in search queries with higher accuracy and elegance, as compared to results achieved by either method alone. DFT suffers from spectral leakage and low resolution at long time periods, whereas autocorrelation fares worse in detecting which periods are significant. Vlachos et al. also define a general similarity measure for periodic signals, which is based on the power spectrum of the respective signals.

(Yang *et al.*, 2003) detect surprising repeating sequences of events by dividing the observations into contiguous chunks of constant length and calculating the information gain of each. Surprising patterns are detected by thresholding the information gain.

(Cao *et al.*, 2007) concentrate on mining periodic location motifs while adaptively discretizing the data via clustering. By performing data duplication based on time slack parameter θ, their method discovers recurring patterns even when they are slightly shifted, distorted, or occur only within a limited time period. Our method goes beyond pattern mining, into actual location prediction.

In the realm of very long-tem predictions (in the order of decades), (Clarke and Gaydos, 1998) analyze a cellular automaton parametrized with historical data that models evolution and spread of urban agglomerations. However, they capture only aggregate, coarse, and high-level patterns, whereas we model fine-grained mobility of individuals. (Jeung *et al.*, 2008) evaluate a hybrid location model that invokes two different prediction algorithms, one for queries that are temporally close, and the other for predictions further into the future. Short-term inference is based on matching recent trajectory patterns, while the long-term alternative focuses on leveraging time periodicity exclusively. However, their approach requires selecting a large number of parameters and metrics (*e.g.*, when to switch from short- to long-term predictor) Additionally, Jeung et al. take only very small amount of actual location data (typically several hours) and synthesize 200 days worth of data. This makes their evaluation limited, especially for the case of long-term predictions. By contrast, we present a unified mobility model that elegantly captures and leverages varying levels of both spatial and temporal resolution. Furthermore, we evaluate on data entirely recorded by real-world sensors.

Recent surge of online social networks sparked interest in predicting people's location

from their online behavior and interactions. (Backstrom *et al.*, 2010) predict home address of Facebook users based on provided addresses of one's friends. An empirically extracted relationship between geographical distance and the probability of friendship between pairs of users is leveraged in order to find a maximum likelihood assignment of addressed to hidden users. The authors show that their method of localizing users is in general more accurate than an IP address-based alternative. However, even their strongest approach captures only a single static home location for each user and the spatial resolution is low. For example, less than 50% of the users are localized within 10 miles of their actual home. Very recently, (Cho *et al.*, 2011) focus on modeling user location in social networks as a dynamic Gaussian mixture, a generative approach postulating that each check-in is induced from the vicinity of either a person's home, work, or is a result of social influence of one's friends. At a larger scale, interesting work by Lee and Sumiya shows it is possible to detect and roughly geo-locate significant social *events* via Twitter by analyzing patterns in people's tweeting behavior (Lee and Sumiya, 2010).

(Hao *et al.*, 2010) augment traditional topic models with location information, which makes it possible to capture topics specific to only certain places (*e.g.*, airports). The inferred topics are subsequently used to enrich travelogues.

(Tao *et al.*, 2004) characterize virtually any motion by expressing location at time t by a polynomial function of n past positions. They show that such a representation can be expressed in recurrent matrix form, and the parameters can be found efficiently. However, Tao et al. concentrate on near-term location prediction based on few consecutive past observations. Their main focus is leveraging the predictions in a client-server architecture resolving spatio-temporal queries.

(Bettencourt and West, 2010) argue for a comprehensive scientific approach to urban planning. They show there are underlying patterns that tie together the size of a city with its emergent characteristics, such as crime rate, number of patents produced, walking speed of its inhabitants, and GDP. The authors argue that cities are the source of many major problems, but also contain the solutions because of their concentrated creativity and productivity.

(Song *et al.*, 2010) take an information-theoretic approach to study the upper and lower bounds on the predictability of people's future location. Borrowing ideas from data compression and applying Fano's inequality, Song et al. show that for their cell phone dataset, location is predictable, on average, 93% of the time when using the most powerful algorithm that performs maximum-likelihood predictions based on historical data. That is, people's mobility appears truly random only 7% of the time. Furthermore, the variance in predictability across subjects is surprisingly small, and even notoriously strong factors, such as age and gender, do not significantly affect the predictability. A generous lower bound on predictability is derived from a model that takes advantage only of information in the bucket of data recorded in the n-th hour of a week when making a prediction about user location at the n-th hour. This isolation of data within buckets precludes any algorithm from leveraging temporal correlations.

(Grenfell *et al.*, 2001) study the dynamics of measles epidemics using wavelet analysis, which allows them to extract spatio-temporal periodic patterns even when they change over time. Furthermore, for each absolute time unit, we know the frequency spectrum that generated it. The approximate phase of the disease trend lines is used to relate different stages of the epidemics to each other.

5.7 Conclusions and Future Work

This work is the first to take on understanding and predicting long-term human mobility in a unified way. We show that it is possible to predict location of a wide variety of hundreds of subjects even years into the future and with high accuracy. We propose and evaluate an efficient and nonparametric model based on eigenanalysis, and demonstrate that it systematically outperforms other strong candidates. Since our model operates over continuous, discrete, and probabilistic data representations, it is quite versatile. Additionally, it has a high predictive as well as descriptive power, since the eigendays capture meaningful patterns in subjects' lives. As our final contribution, we analyze the difficulty of location prediction on a continuum from short- to long-term, and show that Far Out's performance is not significantly affected by the temporal distances.

The cell-based modeling is especially amenable to improvements in future work. Namely, since frequently visited cells have a semantic significance, our probabilistic interpretation can be combined in a Bayesian framework with prior probabilities from large-scale surveys[1] and additional constraints, such as physical limits on mobility, where candidate future locations are strictly constrained by one's current position along with means of transportation available. Finally, it would be interesting to generalize the eigenday approach with a hierarchy of nested eigen-periods, where each level captures only the longer patterns the previous one couldn't (*e.g.*, eigendays→eigenweeks→eigenmonths...).

[1] *e.g.*, American Time Use Survey, National Household Travel Survey

6 Social Network Analysis

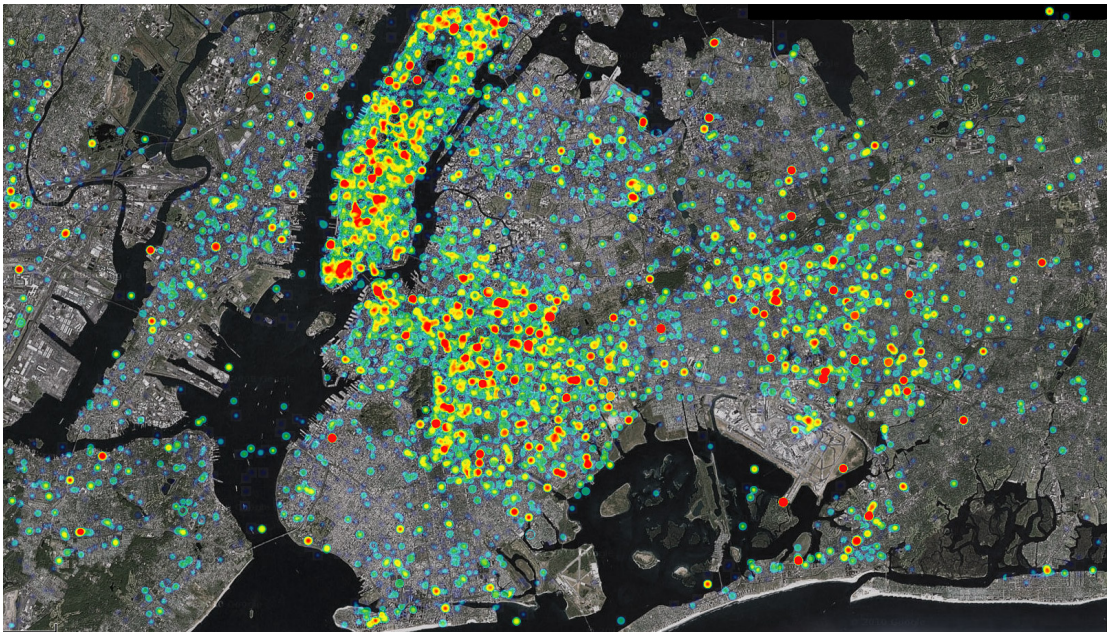


Figure 6.1: A snapshot of a heatmap animation of Twitter users' movement within New York City that captures a typical distribution of geo-tagged messaging on a weekday afternoon. The hotter (more red) an area is, the more people have recently tweeted from that location. Full animation is at http://cs.rochester.edu/u/sadilek/research/.

6.1 Overview

Location plays an essential role in our lives, bridging our online and offline worlds. This chapter explores the interplay between people's location, interactions, and their social ties within a large real-world dataset. We present and evaluate Flap, a system that solves two intimately related tasks: link and location prediction in online social networks. For link prediction, Flap infers social ties by considering patterns in friendship formation, the content of people's messages, and user location. We show that while each component is a weak predictor of friendship alone, combining them results in a strong model, accurately identifying the majority of friendships. For location prediction, Flap implements a scalable probabilistic model of human mobility, where we treat users with known GPS positions as noisy sensors of the location of their friends. We explore supervised and unsupervised learning scenarios, and focus on the efficiency of both learning and inference. We evaluate Flap on a large sample of highly active users from two distinct geographical areas and show that it (1) reconstructs the entire friendship graph with high accuracy even when no edges are given; and (2) infers people's fine-grained location, even when they keep their data private and we can only access the location of their friends. Our models significantly outperform current comparable approaches to either task.

6.2 Motivation

As we noted earlier, our society is founded on the interplay of human relationships and interactions. Since every person is tightly embedded in our social structure, the vast majority of human behavior can be fully understood only in the context of the actions of others. Thus, not surprisingly, more and more evidence shows that when we want to model the behavior of a person, the best predictors are often not based on the person herself, but rather on her friends, relatives, and other *connected* people. For instance, behavioral patterns of people taking taxis, rating movies, choosing cell phone providers, or sharing music are often best predicted by the habits of related people, rather than

by the attributes of the individual such as age, ethnicity, or education (Bell *et al.*, 2007; Pentland, 2008).

Until recently, it was nearly impossible to gather large amounts of data about the connections that play such important roles in our lives. However, this is changing with the explosive increase in the use, popularity, and significance of *online* social media and *mobile* devices.[1] The online aspect makes it practical to collect vast amounts of data, and the mobile element bridges the gap between our online and offline activities. Unlike other computers, phones are aware of the location of their users, and this information is often included in users' posts. In fact, major online social networks are fostering location sharing. Twitter added an explicit GPS tag that can be specified for each tweet (AKA Twitter message update) in early 2010 and is continually improving the location-awareness of its service. Google+, Facebook, FourSquare, and Gowalla allow people to share their location, and to "check-in" at venues. With Google Latitude and Bliin, users can *continually* broadcast their location.

Thus, we now have access to colossal amounts of real-world data containing not just the text and images people post, but also their location. Of course, these three data modalities are not necessarily mutually independent. For instance, photos are often GPS-tagged and locations can also be mentioned, or alluded to, in text.

While the information about users' *location* and *relationships* is important to accurately model their behavior and improve their experience, it is not always available. This chapter explores novel techniques of inferring this latent information from a stream of message updates. We present a unified view on the interplay between people's location, message updates, and their social ties on a large real-world dataset. Our approaches are robust and achieve significantly higher accuracy than the best currently known methods, even in difficult experimental settings spanning diverse geographical areas.

[1] http://www.comscore.com/

6.3 Significance of Results

Consider the task of determining the exact geographic location of an arbitrary user of an online social network. If she routinely geo-tags her posts and makes them public, the problem is relatively easy. However, suppose the location information is hidden, and you only have access to public posts by her friends. By leveraging social ties, our probabilistic location model—the first component of this work—infers where any given user is with high accuracy and fine granularity in both space and time *even when the user keeps his or her posts private*. Since this work shows that once we have location information for some proportion of people, we can infer the location of their friends, one can imagine doing this recursively until the entire target population is covered. To our knowledge, no other work attempts to predict locations in a comparably difficult setting.

The main power of our link prediction approach—the second major component of this work—is that it accurately reconstructs the entire friendship graph even when no "seed" ties are provided. Previous work either obtained very good predictions at the expense of large computational costs (*e.g.*, (Taskar *et al.*, 2003)), thereby limiting those approaches to very small domains, or sacrificed orders of magnitude in accuracy for tractability (*e.g.*, (Liben-Nowell and Kleinberg, 2007; Crandall *et al.*, 2010)). By contrast, we show that our model's performance is comparable to the most powerful relational methods applied in previous work (Taskar *et al.*, 2003), while at the same time being applicable to large real-world domains with tens of millions (as opposed to hundreds) of possible friendships. Since our model leverages users' locations, it not only encompasses virtual friendships, but also begins to tie them together with their real-life groundings.

Prediction of people's location and social ties—especially when considered together—has a number of important applications. They range from improved local content with better social context, through increased security (both personal and electronic) via detection of abnormal behavior tied with one's location, to better organization of one's relationships and connecting virtual friendships with the real-world. We

note that even when friends participate in the same social networking platform, their relationship may not be exposed—either because the connections are hidden or because they have not yet connected online. Flap can also help contain disease outbreaks (Eubank *et al.*, 2004). Our model allows identification of highly mobile individuals as well as their most likely meeting points, both in the past and in the future. These people can be subsequently selected for targeted treatment or preemptive vaccination. Given people's inferred locations, and limited resource budget, a decision-theoretic approach can be used to select optimal emergency policy. Clearly, strong privacy concerns are tied to such applications, as we discuss in the conclusions.

6.4 Related Work

Recent research in **location-based reasoning** explored harnessing data collected on regular smart phones for modeling human behavior (Eagle and Pentland, 2006). Specifically, they model individuals' general location from nearby cell towers and Bluetooth devices at various times of day. Eagle et al. show that predicting if a person is at home, at work, or someplace else can be achieved with more than 90% accuracy. Besides scalability and practicality—social network data is much more readily available than cell phone logs—our work differs in that we include dynamic relational features (location of friends), focus on a finer time granularity, and consider a substantially larger set of locations (hundreds per user, rather than three). Additionally, the observations in our framework (people's self-published locations) are significantly noisier and less regular than cell tower and Bluetooth readings. Finally, our location estimation applies even in situations, where the target people decide to keep their data private.

Backstrom et al. predict the home address of Facebook users based on provided addresses of one's friends (Backstrom *et al.*, 2010). An empirically extracted relationship between geographical distance and the probability of friendship between pairs of users is leveraged in order to find a maximum likelihood assignment of addresses to hidden users. The authors show that their method of localizing users is in general more accurate than an IP address-based alternative. However, even their strongest approach

captures only a single static home location for each user and the spatial resolution is low. For example, less than 50% of the users are localized within 10 miles of their actual home. By contrast, we consider much finer temporal resolution (20 minute intervals) and achieve significantly greater spatial precision, where up to 84% of people's exact dynamic location is correctly inferred.

Very recently, Cho et al. focus on modeling user location in social networks as a dynamic Gaussian mixture, a generative approach postulating that each check-in is induced from the vicinity of either a person's home, work, or is a result of social influence of one's friends (Cho *et al.*, 2011). By contrast, our location model is inherently discrete, which allows us to predict the *exact* location rather than a sample from a generally high-variance continuous distribution; operates at a finer time granularity; and learns the candidate locations from noisy data. Furthermore, our approach leverages the complex temporal and social dependencies between people's locations in a more general, discrete fashion. We show that our model outperforms that of Cho et al. in the experiments presented below.

A number of geolocating applications demonstrate emerging privacy issues in this area. The most impactful ones are arguably Creepy[2], ICanStalkU.com, and PleaseRobMe.com (currently disabled). The purpose of these efforts is to raise awareness about the lack of location privacy in online social networks. Given a username from an online social network, Creepy aggregates location information about the target individual from her GPS-tagged posts and photos, and displays it on a map. ICanStalkU.com scans public Twitter timeline, and extracts GPS metadata from uploaded photos, which often reveal people's current location without them realizing it. PleaseRobMe.com used to extract people's geographic check-ins that imply they are not at home and therefore vulnerable to burglaries.

However, all these applications work only with *publicly available* data. By contrast, this chapter shows that we can *infer* people's precise location even when they keep their data private, as long as some of their friends post their location publicly. Therefore, simply turning off the geolocation features of your phone—which may seem to be a

[2]https://github.com/ilektrojohn/creepy

reliable way to keep your whereabouts secret—does not really protect your privacy unless your friends turn theirs off as well.

While our work concentrates on Twitter users, recent research shows that the predictability of human mobility remains virtually unchanged across a number of demographical attributes such as age and gender (Song *et al.*, 2010). This strongly suggests that our approach achieves similar accuracy for other geographical areas and different samples of users. Finally, we note that although it has been shown that possibly as many as 34% of accounts provide either wrong or misleading symbolic (*e.g.*, city, state) location information in their profiles, our work is largely shielded from this phenomenon since we focus only on raw GPS data that can be more readily verified and is not fed into a geocoder (Hong *et al.*, 2011).

Link prediction is a broad class of extensively studied problems in data mining and graph theory. The general premise is that we want to reconstruct missing edges, or to predict latent or future edges in a graph. Link prediction has important applications in a number of fields including social sciences, physics, and biology. Many variants of link prediction exist that differ in the type of graph modelled (*e.g.*, directed vs. undirected, static vs. evolving), the amount of known existing edges, and the availability of node attributes. In this work, we focus on link prediction in undirected static graphs while leveraging their topological features as well as pairwise attributes of the nodes. For an excellent general overview of computational analysis of social networks at large see (Easley and Kleinberg, 2010).

The problem of **link prediction** has been studied in a large number of domains and contexts; here we mention the ones that are most relevant to our work. Liben-Nowell et al. models the *evolution* of a graph solely from its topological properties (Liben-Nowell and Kleinberg, 2007). The authors evaluate a number of methods of quantifying the probability of a link in large graphs, and conclude that while no single technique is substantially better than the rest, their models outperform a random predictor by a significant margin. This shows that there is important information to be mined from the graph structure alone.

An alternative approach is to leverage the topology as well as attributes of the

individual entities (Taskar *et al.*, 2003). They model friendships of students in an online university community using relational Markov networks. Similarly to our approach, their probabilistic model encompasses a number of features, some of which are based on the attributes of individual users while others model the structure of the friendship graph. Their inference method is standard belief propagation (BP), whereas we develop an efficient and specialized version of BP, which in practice quickly converges. Their domain contains only several hundred candidate social ties. This size restriction is apparently due to the computational challenges posed by their relational model. We, in contrast, consider thousands of individuals who can be connected in arbitrary fashion, which results in tens of millions potential friendships. Furthermore, Taskar et al. assume that some friendships are given to the model at testing time. In this work, we show that it is possible to achieve good performance even with no observed links.

Crandall et al. explore the relationship between co-location of Flickr users and their social ties (Crandall *et al.*, 2010). They model the relationship as an exponential probability distribution and show it fits well to the observed, empirical distribution. They show that the number of distinct places where two users are co-located within various periods of time has the potential to predict a small fraction of the ties quite well. However, the recall is dramatically low. In their Flickr data, only 0.1% of the friendships meet the condition for being predicted with at least 60% confidence. By contrast, with our approach we can predict over 90% of the friendships with confidence beyond 80% (see Figure 6.5). This is consistent with our experiments, where we show that location alone is generally a poor predictor of friendship (consider the "commuter train" example described below on one end of the spectrum, and a pair of friends that never share their location on the other end). We therefore leverage textual similarity and network structure as well, and evaluate the predictive power of our model in terms of AUC while inferring the friendship graph. Additionally, our model does not require setting subtle parameters, such as cell size and time granularity. When we apply the method of Crandall et al. to our Twitter data, the results are (as one would expect) poor; see Figure 6.6 and its analysis in text.

The relationship between social ties and distance has recently received considerable

attention (Liben-Nowell *et al.*, 2005; Backstrom *et al.*, 2010; Scellato *et al.*, 2011b). Even though online interactions are in principle not hampered by physical distance, all works agree that in any social network studied, the probability of friendship generally decreases as the distance between people increases. However, a significant portion of social ties often cannot be explained by location alone. We observe the same pattern in our Twitter data and show that location can be augmented with text and structural features to infer social ties with high accuracy.

Backstrom et al. present a method for predicting links cast as a random walk problem (Backstrom and Leskovec, 2011). A key difference between our approaches is that we can construct the entire social network with high accuracy even when *none* of the edges are observed, whereas Backstrom et al.'s approach depends upon already knowing most of the links in the network along with a set of source and candidate nodes, and only needs to predict relatively few new links. Furthermore, unlike our work, Backstrom et al.'s approach requires many parameters to be selected. In contrast with random walks, approaches related to our belief propagation method for enforcing and chaining soft transitive constraints have been validated in many areas in the machine learning literature, and are implicitly used in many works on link prediction as a way to solve the underlying probabilistic models (Smith and Eisner, 2008; Taskar *et al.*, 2003).

We note that no work to date focused on capturing both directions of the relationship between location and social ties. This chapter concentrates on predicting, in turn, both location and social structure from the remaining data modality.

6.5 The Data

Using the Twitter Search API[3], we collected a sample of public tweets that originated from two distinct geographic areas: Los Angeles (LA) and New York City (NYC). The collection period was one month long and started on May 19 2010. Using a Python script, we periodically queried Twitter with requests of all recent tweets within 150 kilometers of LA's city center, and 100 kilometers within the NYC city center. In order

[3]http://search.twitter.com/api/

to avoid exceeding Twitter's query rate limits and subsequently missing some tweets, we distributed the work over a number of machines with different IP addresses that asynchronously queried the server and merged their results. Twitter does not provide any guarantees as to what sample of existing tweets can be retrieved through their API, but a comparison to official Twitter statistics shows that our method recorded nearly all of the publicly available tweets in those two regions. Altogether, we have logged over 26 million tweets authored by more than 1.2 million unique users (see Table 6.1). To put these statistics in context, the entire NYC metropolitan area has an estimated population of 19 million people.[4] We concentrate on accounts that posted more than 100 GPS-tagged tweets during the one-month data collection period. We refer to them as *geo-active users*. The social network of the 6,237 geo-active users is shown in Fig. 6.2.

6.6 The System: Flap

Flap (Friendship + Location Analysis and Prediction) has three main components responsible for downloading Twitter data, visualization, and learning and inference. The data collection component was described in the previous section. Figure 6.3 shows Flap's visualization a sample of geo-active users in NYC. People are represented by pins on the map and the red links denote friendships (either ground truth or inferred). Beyond standard Google Maps user interface elements, the visualization is controlled via the black toolbar in the upper-right corner. Flap can animate arbitrary segments of the data at various speeds. Selecting a user displays additional information such as his profile, time and text of his recent tweets, and a more detailed map of his current surroundings.

Now we turn to the third—machine learning—module of Flap that has two main tasks. First, it is responsible for learning a model of people's friendships and subsequently revealing hidden friendships. And second, it learns models of users' mobility and predicts their location at any given time. We will now discuss these two tasks and our solutions in turn.

[4] http://www.census.gov/popest/metro/

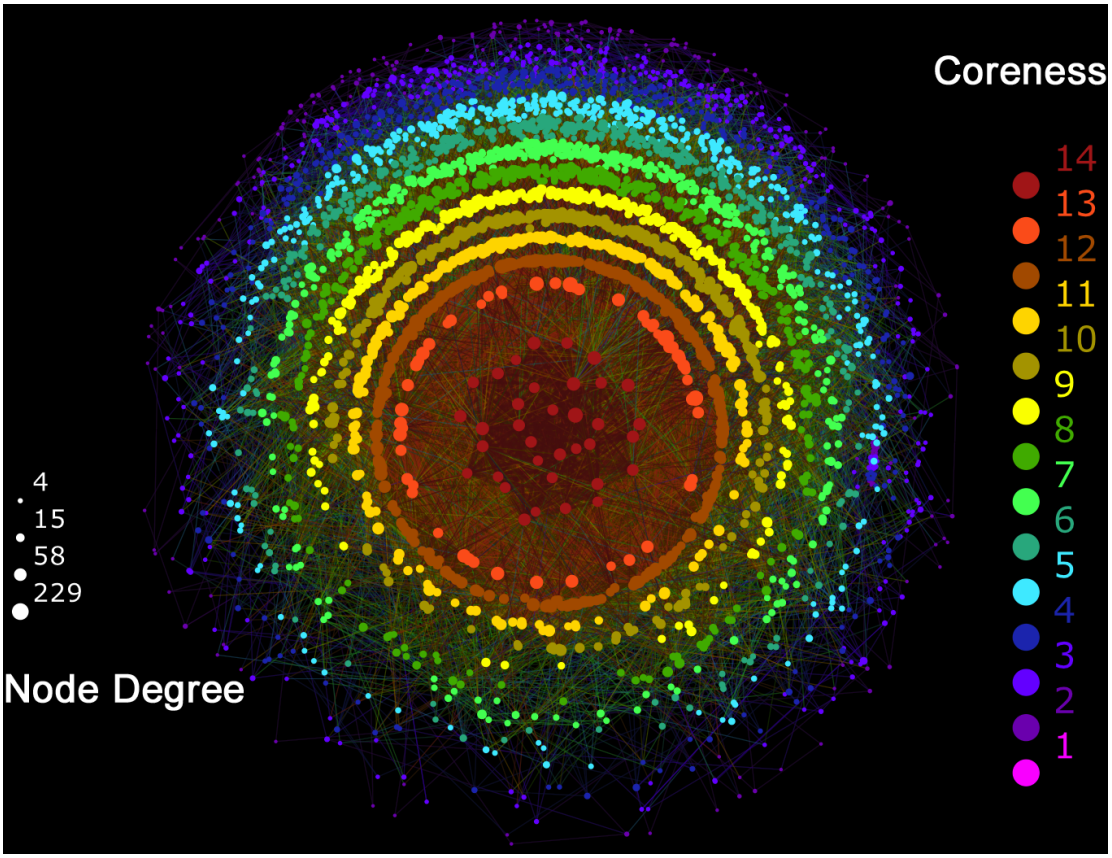


Figure 6.2: Visualization of the social network consisting of the geo-active users. Edges between nodes represent friendships on Twitter. The image has been created using LaNet-vi package implementing k-core decomposition (Beiró *et al.*, 2008). The coreness of nodes is color-coded using the scale on the right. The degree of a node is represented by its size shown on the left. We see that there are relatively few important "hubs" in the central area, and a large number of less connected individuals on the fringes.

New York City & Los Angeles Dataset	
Unique users	1,229,611
Unique geo-active users	11,380
Tweets total	26,118,084
GPS-tagged tweets	7,566,569
GPS-tagged tweets by geo-active users	4,016,286
Unique locations	89,077
Significant locations	25,830
"Follows" relationships between geo-active users	123,182
"Friends" relationships between geo-active users	52,307

Table 6.1: Summary statistics of the data collected from NYC and LA. Geo-active users are ones who geo-tag their tweets relatively frequently (more than 100 times per month). Note that following reciprocity is about 42%, which is consistent with previous findings (Kwak *et al.*, 2010; Java *et al.*, 2007). Unique locations are the result of iterative clustering that merges (on a per-user basis) all locations within 100 meters of each other. Significant location is defined as one that was visited at least five times by at least one person.

6.6.1 Friendship Prediction

The goal of friendship prediction is to reconstruct the entire social graph, where vertices represent users and edges model friendships. We achieve this via an iterative method that operates over the current graph structure and features of pairs of vertices. We first describe the features used by our model of social ties, and then focus on its structure, learning, and inference. In agreement with prior work, we found that no single property of a pair of individuals is a good indicator of the existence or absence of friendship (Liben-Nowell *et al.*, 2005; Cho *et al.*, 2011). Therefore, we combine multiple disparate features—based on text, location, and the topology of the underlying friendship graph.

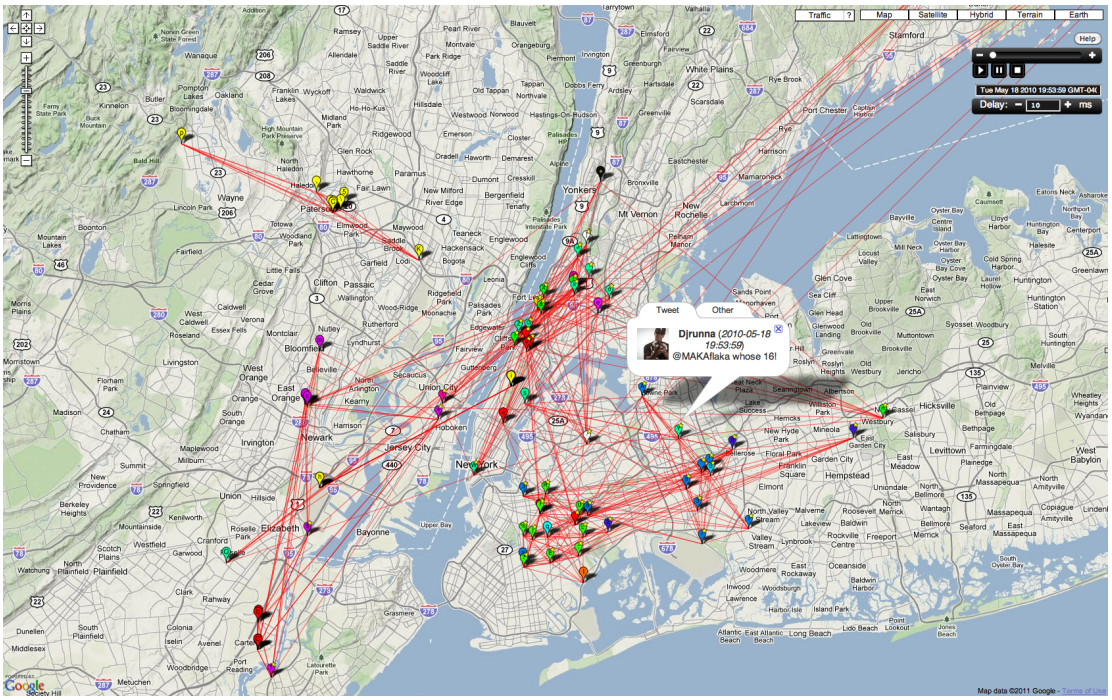


Figure 6.3: Flaps's visualization of a sample of geo-active friends in NYC. Red links between users represent friendships.

Features

The text similarity coefficient quantifies the amount of overlap in the vocabularies of users u and v, and is given by

$$\mathcal{T}(u,v) = \sum_{w \in W(u) \cap W(v) \setminus S} f_u(w) f_v(w), \tag{6.1}$$

where $W(u)$ is the set of words that appear in user u's tweets, S is the set of stop-words (it includes the standard stop words augmented with words commonly used on Twitter, such as RT, im, and lol), and $f_u(w)$ is the frequency of word w in u's vocabulary.

Interestingly, in the Twitter domain, the mentions tags (@) give a clue to user's friendships. However, in the experiments presented here, we *eliminate* all user names that appear in the tweets in order to report results that generalize to other social networks.

Our co-location feature ($\mathcal{C}$) is based on the observation that at least some people who are online friends also meet in the physical world (Gruzd *et al.*, 2011). We make an

assumption that once a user tweets from a location, he or she remains at that location until they tweet again. Even though people generally do not tweet from every single place they visit, this approximate co-location measure still captures how much time pairs of users tend to spend close to each other. The co-location score is given by

$$\mathcal{C}(u,v) = \sum_{\ell_u,\ell_v \in L} \frac{t(\ell_u,\ell_v)}{d(\ell_u,\ell_v)}, \tag{6.2}$$

where L is the union of all locations from which users u and v send messages, $t(\ell_u,\ell_v)$ is the amount of time u spends at location ℓ_u while v is at location ℓ_v. In short, we add up the time overlaps two users spend at their respective locations and we scale each overlap by the distance between the locations. Thus, two individuals spending a lot of common time at nearby places receive a large co-location score, while people who always tweet from two opposite ends of a city have a small co-location score. We have implemented an efficient algorithm that calculates $\mathcal{C}(u,v)$ for a pair of users in time $O(n)$ where n is the minimum number of GPS-tagged messages created by either user u or v. Note that unlike previous work (*e.g.*, (Crandall *et al.*, 2010; Backstrom and Leskovec, 2011)), our co-location feature is continuous and does not require discretization, thresholding, or parameter selection.

As a graph structure feature, we use the meet/min coefficient ($\mathcal{M}$) and its generalized version ($\mathcal{M}_{\mathbb{E}}$) defined in equations 6.3 and 6.4 respectively.

$$\mathcal{M}(u,v) = \frac{\big|N(u) \cap N(v)\big|}{\min\Big(\big|N(u)\big|, \big|N(v)\big|\Big)} \tag{6.3}$$

$$\mathcal{M}_{\mathbb{E}}(u,v) = \frac{\sum\limits_{n \in N(u) \cap N(v)} p_{nu} p_{nv}}{\min\Big(\sum\limits_{n \in N(u)} p_{nu}, \sum\limits_{n \in N(v)} p_{nv}\Big)} \tag{6.4}$$

$N(u)$ is the set of neighbors of node u and p_{nu} is the probability of edge (n,u). The standard meet/min coefficient counts the number of common neighbors of u and v (this quantity is equal to the number of triads that the edge (u,v) would complete, an important measure in structural balance theory (Easley and Kleinberg, 2010)), and

scales by the size of the neighborhood of either u or v, whichever is smaller. Intuitively, $\mathcal{M}(u, v)$ expresses how extensive is the overlap between friendlists of users u and v with respect to the size of the shorter friendlist. The expectation of the meet/min coefficient $\mathcal{M}_{\mathbb{E}}$ calculates the same quantities but in terms of their expected values on a graph where each edge is weighted by its probability. Neither measure depends on the existence or probability of edge (u, v) itself.

Since the $\mathcal{T}$ and $\mathcal{C}$ scores are always observed, we use a regression decision tree to unify them, in a pre-processing step, into one feature $\mathcal{DT}(u, v)$, which is the decision tree's prediction given $\mathcal{T}(u, v)$ and $\mathcal{C}(u, v)$. Thus, we end up with one feature function for the observed variables ($\mathcal{DT}$) and one for the hidden variables ($\mathcal{M}_{\mathbb{E}}$).

We have experimented with other features, including the Jaccard coefficient, preferential attachment, hypergeometric coefficient, and others. However, our work is motivated by having an efficient and scalable model. A decision tree-based feature selection showed that our three measures ($\mathcal{T}$, $\mathcal{C}$, and $\mathcal{M}_{\mathbb{E}}$) jointly represent the largest information value. Finally, while calculating the features for all pairs of n users is an $O(n^2)$ operation, it can be significantly sped up via locality-sensitive hashing (Datar *et al.*, 2004).

Learning and Inference

Our probabilistic model of the friendship network is a Markov random field that has a hidden node for each possible friendship. Since the friendship relationship is symmetric and irreflexive, our model contains $n(n-1)/2$ hidden nodes, where n is the number of users. Each hidden node is connected to an observed node ($\mathcal{DT}$) and to all other hidden nodes.

Ultimately, we are interested in the probability of existence of an edge (friendship) given the current graph structure and the pairwise features of the vertices (users) the edge is incident on. Applying Bayes' theorem while assuming mutual independence of

features $\mathcal{DT}$ and $\mathcal{M}_{\mathbb{E}}$, we can write

$$\begin{aligned} &P(E=1|DT=d, M_{\mathbb{E}}=m) = \\ &= P(DT=d|E=1)P(E=1|M_{\mathbb{E}}=m)/Z \end{aligned} \tag{6.5}$$

where

$$Z = \sum_{i \in \{0,1\}} P(DT=d|E=i)P(E=i|M_{\mathbb{E}}=m).$$

E, DT, and $M_{\mathbb{E}}$ are random variables that represent edge existence, $\mathcal{DT}$ score, and $\mathcal{M}_{\mathbb{E}}$ score, respectively. In equation 6.5, we applied the equality

$$P(M_{\mathbb{E}}|E) = P(E|M_{\mathbb{E}})P(M_{\mathbb{E}})/P(E)$$

and subsequent simplifications so that we do not need to explicitly model $P(E)$.

At learning time, we first train a regression decision tree $\mathcal{DT}$ and prune it using ten-fold cross-validation to prevent overfitting. We also perform maximum likelihood learning of the parameters $P(DT|E)$ and $P(E|M_{\mathbb{E}})$. We chose the decision tree pre-processing step for several reasons. First, the text and location-based features considered individually or independently have very poor predictive power. Therefore, models such as logistic regression tend to have low accuracy. Furthermore, the relationships between the observed attributes of a pair of users and the their friendship is often quite complex. For example, it is not simply the case that a friendship is more and more likely to exist as people spend larger and larger amounts of time near each other. Consider two strangers that happen to take the same train to work, and tweet every time it goes through a station. Our dataset contains a number of instances of this sort. During the train ride, their co-location could not be higher and yet they are not friends on Twitter. This largely precludes success of classifiers that are looking for a simple decision surface.

At inference time, we use $\mathcal{DT}$ to make preliminary predictions on the test data. Next, we execute a customized loopy belief propagation algorithm that is initialized with the probabilities estimated by $\mathcal{DT}$ (see Algorithm 1). Step 6 is where an edge receives belief updates from the other edges as well as the $\mathcal{DT}$ prior.

Belief propagation (BP) is a family of message passing algorithms that perform inference in graphical models. BP is applicable to all models discussed above. BP

iinterplays proven to be exact and to converge for certain classes of graphs, such as trees, but its behavior on general cyclic graphs is poorly understood (Pearl, 1988). However, in many practical applications, BP performs surprisingly well (Murphy *et al.*, 1999).

Even though the graphical model is dense, our algorithm converges within several hundred iterations, due in part to the sufficiently accurate initialization and regularization provided by the decision tree. Note that the algorithm can also function in an online fashion: as new active users appear in the Twitter public timeline, they are processed by the decision tree and added to Q. This is an attractive mode, where the model is always up to date and takes advantage of all available data.

Algorithm 3 : refineEdgeProbabilities(Q)

Input: Q: list containing all potential edges between pairs of vertices along with their preliminary probabilities

Output: Q: input list Q with refined probabilities

1: **while** Q has not converged **do**
2: sort Q high to low by estimated edge probability
3: **for each** $\langle e, P(e)\rangle$ **in** Q **do**
4: $dt \Leftarrow \mathcal{DT}(e)$
5: $m \Leftarrow \mathcal{M}_{\mathbb{E}}(e)$
6: $P(e) \Leftarrow \frac{P(DT{=}dt|E{=}1)P(E{=}1|\mathcal{M}_{\mathbb{E}}{=}m)}{\sum_{i\in\{0,1\}} P(DT{=}dt|E{=}i)P(E{=}i|\mathcal{M}_{\mathbb{E}}{=}m)}$
7: **end for**
8: **end while**
9: **return** Q

6.6.2 Location Prediction

The goal of Flap's location prediction component is to infer the most likely location of person u at any time. The input consists of a sequence of locations visited by u's friends (and for supervised learning, locations of u himself over the training period),

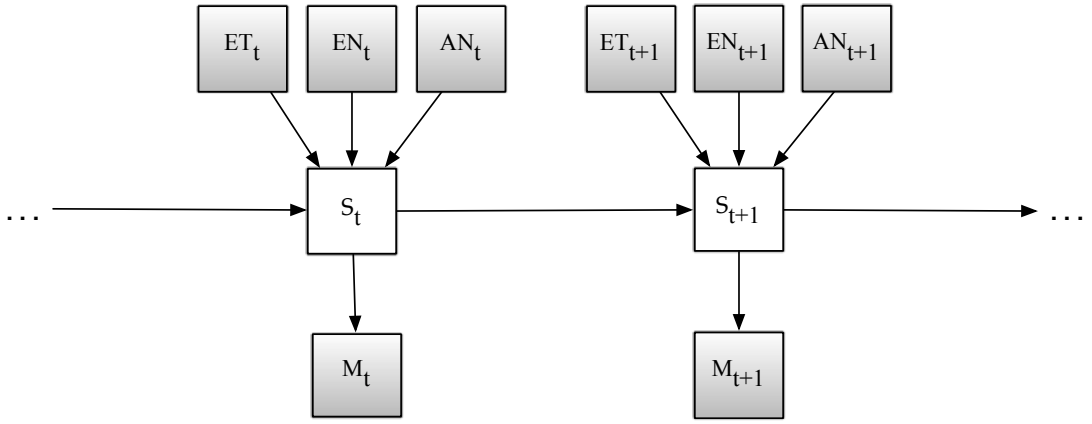


Figure 6.4: Two consecutive time slices of our dynamic Bayesian network for modeling motion patterns of Twitter users from n friends. All nodes are discrete, shaded nodes represent observed random variables, unfilled denote hidden variables.

along with corresponding time information. The model outputs the most likely sequence of locations u visited over a given time period.

We model user location in a dynamic Bayesian network shown in Figure 6.4. In each time slice, we have one hidden node and a number of observed nodes, all of which are discrete. The hidden node represents the location of the target user (u). The node td represents the time of day and w determines if a given day is a work day or a free day (weekend or a national holiday). Each of the remaining observed nodes ($f1$ through fn) represents the location of one of the target user's friends. Since the average node degree of geo-active users is 9.2, we concentrate on $n \in \{0, 1, 2, \ldots, 9\}$, although our approach works for arbitrary nonnegative values of n. Each node is indexed by time slice.

The domains of the random variables are generated from the Twitter dataset in the following way. First, for each user, we extract a set of distinct locations they tweet from. Then, we iteratively merge (cluster) all locations that are within 100 meters of each other in order to account for GPS sensor noise, which is especially severe in areas with tall buildings, such as Manhattan. The location merging is done separately for each user and we call the resulting locations *unique*. We subsequently remove all merged locations that the user visited fewer than five times and assign a unique label to each remaining place. These labels are the domains of u and fi's. We call such places

significant.

The above place indexing yields a total of 89,077 unique locations, out of which 25,830 were visited at least five times by at least one user. There were 2,467,149 tweets total posted from the significant locations in the 4 week model evaluation period. Table 6.1 lists summary statistics.

We model each person's location in 20 minute increments, since more than 90% of the users tweet with lower frequency. Therefore, the domain of the time of day random variable t_d is $\{0, \ldots, 71\}$ (total of 24/0.3 time intervals in any given day).

Learning

We explore both supervised and unsupervised learning of user mobility. In the earlier case, for each user, we train a DBN on the first three weeks of data with known hidden location values. In the latter case, the hidden labels are unknown to the system.

During *supervised* learning, we find a set of parameters (discrete probability distributions) θ that maximize the log-likelihood of the training data. This is achieved by optimizing the following objective function.

$$\theta^\star = \underset{\theta}{\operatorname{argmax}} \ \log\big(\Pr(x_{1:t}, y_{1:t}|\theta)\big), \tag{6.6}$$

where $x_{1:t}$ and $y_{1:t}$ represent the sequence of observed and hidden values, respectively, between times 1 and t, and $\theta^\star$ is the set of optimal model parameters. In our implementation, we represent probabilities and likelihoods with their log-counterparts to avoid arithmetic underflow.

For *unsupervised* learning, we perform expectation-maximization (EM) (Dempster *et al.*, 1977). In the E step, the values of the hidden nodes are inferred using the current DBN parameter values (initialized randomly). In the subsequent M step, the inferred values of the hidden nodes are in turn used to update the parameters. This process is repeated until convergence, at which point the EM algorithm outputs a maximum

likelihood point estimate of the DBN parameters. The corresponding optimization problem can be written as

$$\theta^\star = \underset{\theta}{\operatorname{argmax}} \ \log \sum_{y_{1:t}} \Pr\big(x_{1:t}, y_{1:t}|\theta)\big), \tag{6.7}$$

where we sum over all possible values of hidden nodes $y_{1:t}$. Since equation 6.7 is computationally intractable for sizable domains, we simplify by optimizing its lower bound instead, similar to (Ghahramani, 1998).

The random initialization of the EM procedure has a profound influence on the final set of learned parameter values. As a result, EM is prone to getting "stuck" in a local optimum. To mitigate this problem, we perform deterministic simulated annealing (Ueda and Nakano, 1998). The basic idea is to reduce the undesirable influence of the initial random set of parameters by "smoothing" the objective function so that it hopefully has fewer local optima. Mathematically, this is written as

$$\theta^\star(\tau_1, \ldots, \tau_m) = \underset{\theta}{\operatorname{argmax}} \ \tau_i \log \sum_{y_{1:t}} \Pr\big(x_{1:t}, y_{1:t}|\theta\big)^{\frac{1}{\tau_i}}. \tag{6.8}$$

Here, $\tau_1, \ldots, \tau_m$ is a sequence of parameters, each of which corresponds to a different amount of smoothing of the original objective function (shown in equation 6.7). The sequence is often called a *temperature schedule* in the simulated annealing literature, because equation 6.8 has analogs to free energy in physics. Therefore, we start with a relatively high temperature τ_1 and gradually lower it until $\tau_m = 1$, which recovers the original objective function.

Inference

At inference time, we are interested in the most likely explanation of the observed data. That is, given a sequence of locations visited by one's friends, along with the corresponding time and day type, our model outputs the most likely sequence of locations one visited over the given time period.

Flap runs a variant of Viterbi decoding to efficiently calculate the most likely state of the hidden nodes. In our model, Viterbi decoding is given by

$$y^\star_{1:t} = \underset{y_{1:t}}{\operatorname{argmax}} \ \log \big(\Pr(y_{1:t}|x_{1:t})\big), \tag{6.9}$$

where $\Pr(y_{1:t}|x_{1:t})$ is conditional probability of a sequence of hidden states $y_{1:t}$ given a concrete sequence of observations $x_{1:t}$ between times 1 and t.

In each time slice, we coalesce all observed nodes with their hidden parent node, and since we have one hidden node in each time slice, we apply dynamic programming and achieve polynomial runtimes in a way similar to (Jordan, 1998). Specifically, the time complexity of our inference is $O(T\,|Y|^2)$, where T is the number of time slices and Y is the set of possible hidden state values (potential locations).

Therefore, the overall time complexity of learning and inference for any given target user is $O(kT\,|Y|^2)$, where k is the number of EM iterations ($k = 1$ for supervised learning). This renders our model tractable even for very large domains that evolve over long periods of time with fine granularity. Next, we turn to our experiments, and analysis of results.

6.7 Evaluation

For clarity, we discuss experimental results for each of the two Flap's tasks separately.

6.7.1 Friendship Prediction

We evaluate Flap on friendship prediction using two-fold cross-validation in which we train on LA and test on NY data, and vice versa. We average the results over the two runs. We varied the amount of randomly selected edges provided to the model at testing time from 0 to 50%.

Flap reconstructs the friendship graph well over a wide range of conditions—even when given no edges (Figure 6.5 and Table 6.2). It far outperforms the baseline model (decision tree) and the precision/recall breakeven points are comparable to those of (Taskar *et al.*, 2003), even though our domain is orders of magnitude larger and our model is more tractable.

We also compare our model to that of Crandall et al. (Crandall *et al.*, 2010), summarized in Section 6.4. Figure 6.6 shows the results of their contemporaneous events

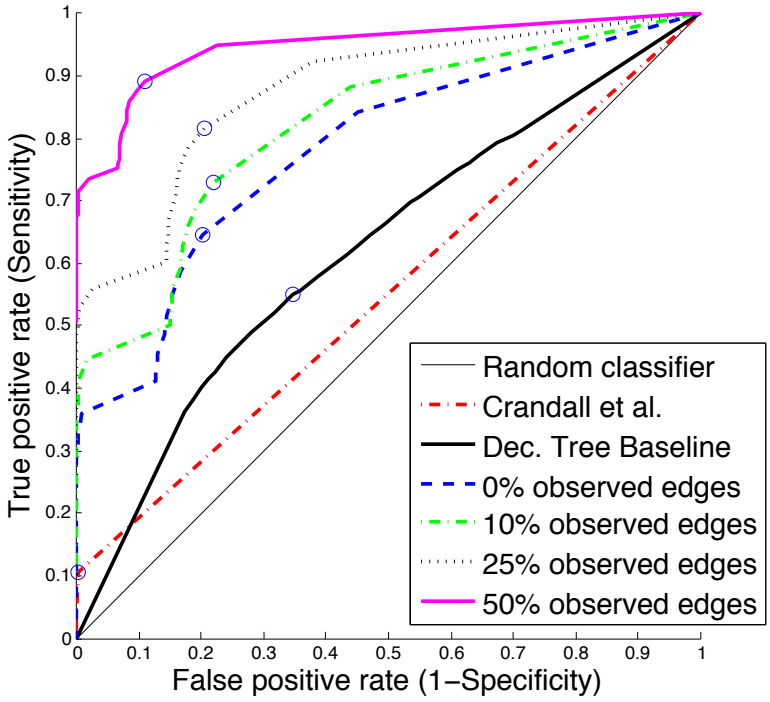


Figure 6.5: Averaged ROC curves for decision tree baseline, Crandall et al.'s model with the most favorable setting of parameters ($s = 0.001$ and $t = 4$ hours), and Flap.

counting procedure on our Twitter data for various spatial and temporal resolutions. We see that in our dataset, the relationship between co-location and friendship is much more complex and non-monotonic as compared to their Flickr dataset. As a result, the predictive performance of Crandall et al.'s model on our data is poor. When probabilistically predicting social ties based on the number of contemporaneous events, the accuracy is 0.001%, precision 0.008, and recall 0.007 (in the best case, where $s = 0.001$ and $t = 4$ hours). There are two conclusions based on this result. First, similarly to Liben-Nowell et al. (Liben-Nowell *et al.*, 2005), we observe that geographic distance alone is not sufficient to accurately model social ties. And second, looking at the performance of (Crandall *et al.*, 2010)'s approach on the Flick data comprising the entire world, versus its performance on our LA and NYC data, we see that inferring relationships from co-location data in dense and relatively small geographical areas can be a more challenging task. This is important, as the majority of population lives and interacts in such metropolitan areas. However, our work shows that when we leverage

	#E	0%	10%	25%	50%
AUC Flap	6.5×10^7	0.78	0.82	0.88	0.95
AUC Crandall et al.	6.5×10^7	0.55	-	-	-
P=R Flap	6.5×10^7	0.28	0.36	0.47	0.64
P=R Crandall et al.	6.5×10^7	0.05	-	-	-
P=R Taskar et al.	$\sim 4 \times 10^2$	N/A	0.47	0.58	0.73

Table 6.2: Summary of model evaluation. The #E column represents the number of candidate edges that exist in the social graph. The remaining columns denote the proportions of friendships given to the models at testing time. AUC is the area under the ROC curve; P=R denotes precision/recall breakeven points. All results are based on our Twitter dataset, except for the P=R results for Taskar et al., which are based on their—much smaller—university dataset as their model does not scale to larger networks; see text for details.

additional information channels beyond co-location, and embed them in a probabilistic model, we can infer social ties quite well.

In order to explore how our model performs in the context of *strong* ties, in both LA and NYC, we selected a subgraph that contains only active users who are members of a clique of size at least eight. We again evaluated via cross-validation as above. Flap reconstructs the friendship network of the 83 people with 0.92 precision and 0.85 recall, whereas the baseline decision tree achieves precision of 0.83 and recall of 0.51. Interestingly, the co-location feature plays a major role here because the cliques of friends spend most of their time in relatively small areas.

6.7.2 Location Prediction

Our evaluation is done in terms of *accuracy*—the percentage of timeslices for which the model infers the correct user location. We have a separate dynamic Bayesian network model for each user.

In order to evaluate the models learned in a *supervised* fashion, we train each model

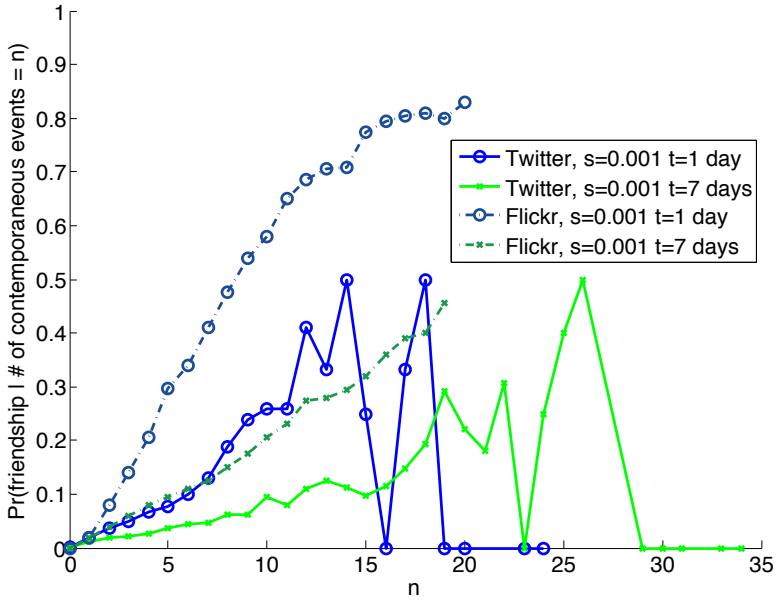


Figure 6.6: Comparison of the intensity of co-location of pairs of users versus the probability of their friendship in our Twitter and Crandall et al.'s Flickr datasets. We see that the relationship is more complex on Twitter, causing a simple model of social ties to achieve very low predictive accuracy. s is the size of cells in degrees in which we count the co-located events and t is the time slack; compare with Figure 2 in (Crandall *et al.*, 2010).

on three weeks worth of data (5/19/2010 00:00:00 – 6/8/2010 23:59:59) and test on the following fourth week (6/9/2010 00:00:00 – 6/15/2010 23:59:59). We always use the respective local time: PDT for LA, and EDT for NYC. We vary the number of friends (n) that we harness as sensors for each individual from 0 to 9. We always use the n most geo-active friends, and introduce a special missing value for users who have fewer than n friends. We evaluate the overall performance via cross-validation. In each fold of cross-validation, we designate a target user and run learning and inference for him. This process is repeated for all users, and we report the average results over all runs for a given value of n (Figure 6.7).

For models learned in an *unsupervised* manner, we also apply cross-validation as above. The hidden locations are learned via unsupervised clustering as described above. The temperature schedule for the EM procedure is given by $\tau_{i+1} = \tau_i \times 0.8$, with initial

temperature $\tau_1 = 10$ (see equation 6.8). This results in calculating the likelihood at 11 different temperatures for each EM run. The EM procedure always converged within one thousand iterations, resulting in runtimes under a minute per user even in the largest domain.

We compare the results obtained by our DBN models to random and naïve baselines, and to the currently *strongest* mobility model of Cho et al. (Cho *et al.*, 2011). The random model is given the number of hidden locations for each user and guesses target user's location uniformly at random for each time slice. The naïve model always outputs the location at which the target user spends most of his time in the training data. We consider a prediction made by Cho et al.'s model accurate if it lies within 100 meters (roughly a city block) of the true user location.

Figure 6.7 summarizes the results. As expected, the supervised models perform better than their unsupervised counterparts. However, given the complexity of the domain and the computational efficiency of our models during training as well as testing, even the unsupervised models achieve respectable accuracy. The DBN approaches are significantly better than both random and naïve baselines, and they also dominate (Cho *et al.*, 2011)'s social mobility model (PSMM) by a large margin. We believe this is mainly because people's mobility in condensed metropolitan areas often does not nicely decompose into "home" and "work" states, and the social influence on user location is not simply an attractive force (*i.e.*, one does not necessarily tend to appear closer to one's friends). For instance, consider two co-workers, one having a morning shift and the other a night shift in the same store. Their mobility is certainly intertwined, but the force between their location is a repulsive one, as when the first one is working, the other is sleeping at home. Unlike other approaches, our DBN model correctly learns such nonlinear patterns (both temporal and social).

The results are very encouraging. For example, even when the model is given information only about one's two friends, and no information about the target user (Unsupervised, $n = 2$ in Figure 6.7), it infers the correct location 47% of the time. As we increase the number of available friends to nine (Unsupervised, $n = 9$), we achieve 57% accuracy. When historical data about the mobility of the target user and his friends is

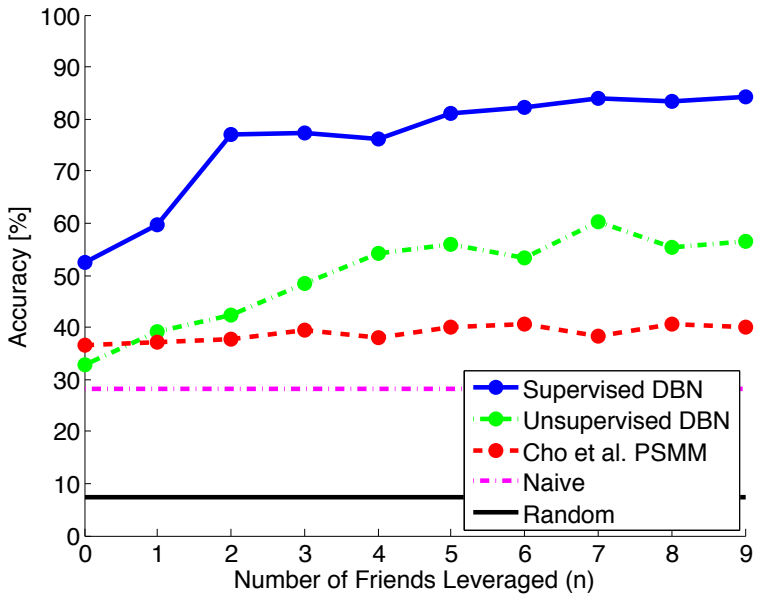


Figure 6.7: Predictive accuracy of location models. The performance of the two baseline models is by design independent of number of friends considered.

available, we can estimate the correct location 77% of the time when two friends are available (Supervised, $n = 2$) and 84.3% with nine friends (Supervised, $n = 9$). As n increases, the accuracy generally improves. Specifically, we see that there is a significant boost from $n = 0$ to $n = 2$, after which the curves plateau. This suggests that a few active friends explain one's mobility well. We also see that simply outputting the most commonly visited location (Naïve) yields poor results since people tend to lead fairly dynamic lives.

6.8 Conclusions and Future Work

Location information linked with the content of users' messages in online social networks is a rich information source that is now accessible to machines in massive volumes and at ever-increasing real-time streaming rates. This data became readily available only very recently. In this work, we show that there are significant patterns that characterize locations of individuals and their friends. These patterns can be leveraged in probabilistic models that infer people's locations as well as social ties with high accuracy. Moreover,

the prediction accuracy degrades gracefully as we limit the amount of observed data available to the models, suggesting successful future deployment of Flap at a scale of an entire social network.

Our approach is quite powerful, as it allows us to reason even about the location of people who keep their messages and GPS data private, or have disabled the geo-features on their computers and phones altogether. Furthermore, unlike all existing approaches, our model of social ties reconstructs the entire friendship network with high accuracy even when the model is not "seeded" with a sample of known friendships. At the same time, we show that the predictions improve as we provide more observed edges at testing time.

By training the model on one geographical area and testing on the other using cross-validation (total of 4 million geo-tagged public tweets we collected from Los Angeles and New York City metropolitan areas), we show that Flap discovers robust patterns in the formation of friendships that transcend diverse and distant areas of the USA. We conclude that no single property of a pair of individuals is a good indicator of the existence or absence of friendship. And no single friend is a good predictor of one's location. Rather, we need to combine multiple disparate features—based on text, location, and the topology of the underlying friendship graph—in order to achieve good performance.

In our current work, we are extending the model to leverage the textual content of the tweets, as it contains hints about locations that are not captured by our existing features. We are currently exploring language understanding and toponym resolution techniques vital for tapping this information. We also focus on casting the two problems explored in this chapter in a unified formalism and solving them *jointly*, perhaps in a recursive fashion.

We recognize that there are substantial ethical questions ahead, specifically concerning tradeoffs between the values our automated systems create versus user privacy. For example, our unsupervised experiments show that location can be inferred even for people who keep their tweets and location private, and thus may believe that they are "untrackable." These issues will need to be addressed in parallel with the development

of our models. Other researchers have started exploring solutions to privacy concerns through data obfuscation (Brush *et al.*, 2010).

However, we believe that the benefits of Flap—in helping to connect and localize users, and in building smarter systems—outweigh the possible dangers. There are many exciting practical applications that have the potential to change people's lives that rely on location and link estimation. These include context-aware crowdsourcing, timely and accurate recommendations, better local content, disease prevention and containment, security, traffic modeling, emergency response, and others. In the next chapter, we will discuss applications of our models in the public health domain, where we show that social media can be used to reason and predict disease epidemics with little delay and with fine granularity.

7 Fine-Grained Computational Epidemiology

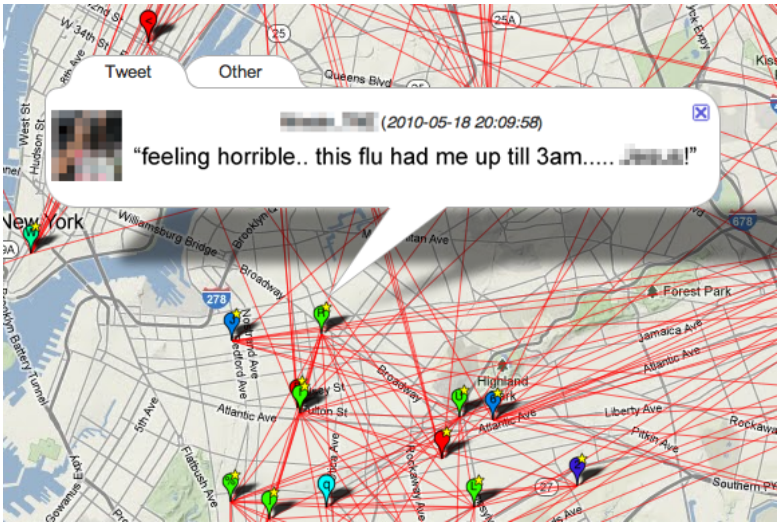


Figure 7.1: Visualization of a sample of friends in New York City. The red links between users represent friendships, and the colored pins show their current location on a map. We see the highlighted person X complaining about her health, and hinting about the specifics of her ailment. This chapter investigates to what extent can we predict the day-to-day health of individuals by considering their physical encounters and social interactions with people like X.

7.1 Introduction

Given that five of your friends have flu-like symptoms, and that you have recently met eight people, possibly strangers, who complained about having runny noses and headaches, what is the probability that you will soon become ill as well? This chapter explores how accurately such questions can be answered across a large sample of people participating in online social media (see Fig. 7.1).

Imagine Joe is about to take off on an airplane and quickly posts a Twitter update from his phone. He writes that he has a fever and feels awful. Since Joe has a public Twitter profile, we know who some of his friends are, and from his GPS-tagged messages we see some of the places he has recently visited. Additionally, we can infer a large fraction of the hidden parts of Joe's social network and his latent locations by applying the results of previous work, as we discuss below. In the same manner, we can identify other people who are likely to be at Joe's airport, or even on the same flight. Using both the observed and inferred information, we can now monitor individuals who likely came into contact with Joe, such as the passengers seated next to him. Joe's disease may have been transmitted to them, and vice versa, though they may not exhibit any symptoms yet. As people travel to their respective destinations, they may be infecting others along the way. Eventually, some of the people will tweet about how they feel, and we can observe at least a fraction of the population that actually contracted the disease.

The example just given illustrates our vision of how public health modeling may look like in the near future. In this chapter, we report on our initial progress towards making this vision a reality.

Research in computational epidemiology to date has concentrated on coarse-grained statistical analysis of populations, often synthetic ones. By contrast, our work focuses on fine-grained modeling of the spread of infectious diseases throughout a large real-world social network. Specifically, we study the roles that social ties and interactions between specific individuals play in the progress of a contagion. We focus on public Twitter data, where we find that for every health-related message there are more than 1,000 unrelated

ones. This class imbalance makes classification particularly challenging. Nonetheless, we present a framework that accurately identifies sick individuals from the content of online communication. Evaluation on a sample of 2.5 million geo-tagged Twitter messages shows that social ties to infected, symptomatic people, as well as the intensity of recent co-location, sharply increase one's likelihood of contracting the illness in the near future. To our knowledge, this work is the first to model the interplay of social activity, human mobility, and the spread of infectious disease in a large real-world population. Furthermore, we provide the first quantifiable estimates of the characteristics of disease transmission on a large scale without active user participation—a step towards our ability to model and predict the emergence of global epidemics from day-to-day interpersonal interactions.

Traditionally, public health is monitored via surveys and by aggregating statistics obtained from healthcare providers. Such methods are costly, slow, and may be biased. For instance, a person with flu is recorded only after he or she visits a doctor's office and the information is sent to the appropriate agency. Affected people who do not seek treatment, or do not respond to surveys are virtually invisible to the traditional methods.

Recently, digital media has been successfully used to significantly reduce the latency and improve the overall effectiveness of public health monitoring. Perhaps most notably, Google Flu Trends[1] models the prevalence of flu via analysis of geo-located search queries (Ginsberg *et al.*, 2008). Other researchers leverage news articles and associated user discussions to monitor infectious diseases (Cui *et al.*, 2011).

Twitter itself has been recently shown to accurately assess the overall prevalence of flu independently in a number of countries with accuracy comparable to current state of the art methods including Google Flu and Center for Disease Control and Prevention (CDC) statistics[2] (Lampos *et al.*, 2010; Culotta, 2010; Signorini *et al.*, 2011). However, even the state of the art systems suffer from two major drawbacks. First, they produce only coarse, aggregate statistics, such as the expected number of people afflicted by flu

[1] http://www.google.org/flutrends/

[2] http://www.cdc.gov/datastatistics/

in Texas. Furthermore, they often perform mere passive monitoring, and prediction is severely limited by the low resolution of the aggregate approach.

By contrast, our work takes a bottom-up approach, where we take into account the fine-grained interactions between individuals. We apply machine learning techniques to the difficult task of detecting ill individuals based on the content of their Twitter status updates. We are then able to estimate the physical interactions between healthy and sick people via their online activities, and model the impact of these interactions on public health.

Recent work has demonstrated that micro-blogging data can be used to predict a variety of phenomena, including movie box-office revenues (Asur and Huberman, 2010), elections (Tumasjan *et al.*, 2010), and flu epidemics (Lampos *et al.*, 2010). Most research to date has focused on predicting aggregate properties of the population from the activity of the bloggers. A different kind of problem one can pose, however, is to predict the behavior or state of *particular individuals* within the social network. For instance, one could try to predict whether a person will go to a movie or vote for a particular candidate based on micro-blog data. The individual's own data may or may not be accessible. At one extreme, the task is to predict his behavior or state by considering only data from other people. For example, Sadilek *et al.* (2012a) show that a person's location can be predicted with a high degree of accuracy based on only the geo-tagged posts (a.k.a. tweets) of his friends on Twitter.

This chapter explores fine-grained prediction of the *health* of individuals on the basis of such social network data—an important instance of the general problem of modeling dynamic properties of participants in large real-world social networks. We begin by building upon previous work on classification of health-related text messages (Culotta, 2010; Paul and Dredze, 2011a; Sadilek *et al.*, 2012b), to learn a robust SVM classifier that infers the health state of a person based on the content of his tweets. We then learn a conditional random field (CRF) model that *predicts* an individual's health status, using features derived from the tweets and locations of other people. Performance of the CRF is significantly enhanced by including features that are not only based on the health status of friends, but are also based on the estimated encounters with already

sick, symptomatic individuals in the dataset, including non-friends. Thus, the model is able to capture the role of *location* in the spread of an infectious disease, the impact of the *duration* of co-location on disease transmission, as well as the *delay* between a contagion event and the onset of the symptoms. Using the Viterbi algorithm to infer the most likely sequence of a subject's health states over time, we are able to predict the days a person is ill with 0.94 precision and 0.18 recall. These results far outperform alternative models.

This work is an important step towards the development of automated methods that identify disease vectors, trace the transmission between concrete individuals, and ultimately help us understand and predict the spread of infectious diseases with fine granularity. It provides a foundation for research on fundamental questions of public health, such as: What roles do co-location and social ties play in the spread of infectious diseases from person to person? How does an epidemic on a population scale emerge from low-level interactions between people in the course of their everyday lives? Can we identify a potentially non-cooperative individual who is a vector of a dangerous disease, *i.e.*, a "Typhoid Mary"? What is the interaction between friendship, location, and co-location in the spread of communicable diseases?

Our results also prove useful for deploying sickness prevention resources, and for applications that help an individual maintain his or her health. For example, a person predicted to be at high risk of the flu could be specifically encouraged to get the flu vaccine, and recommendations can be made about which places pose a high risk of getting infected. Finally, the kinds of models we explore are not limited to the health domain. For instance, a recent study showed that the spread of cheating behavior in online computer games exhibits patterns similar to the spread of a disease (Blackburn *et al.*, 2011). The close relationship between the spread of disease and information in general is well known (Easley and Kleinberg, 2010). By changing the mapping from text to features, the same approach can be used to model and predict the transmission of political ideas, purchasing preferences, or practically any other behavioral phenomena.

7.2 The Data

We use a subset of the dataset introduced in the previous chapter. We briefly review it here. Using the Twitter Search API[3], we collected a sample of public tweets that originated from the New York City (NYC) metropolitan area shown in Fig. 6.1. The collection period was one month long and started on May 18, 2010. Using a Python script, we periodically queried Twitter for all recent tweets within 100 kilometers of the NYC city center. In order to avoid exceeding Twitter's query rate limits and subsequently missing some tweets, we distributed the work over a number of machines with different IP addresses that asynchronously queried the server and merged their results. Twitter does not provide any guarantees as to what sample of existing tweets can be retrieved through their API, but a comparison to official Twitter statistics shows that our method recorded the majority of the publicly available tweets in the region. Altogether, we have logged nearly 16 million tweets authored by more than 630 thousand unique users (see Table 6.1). To put these statistics in context, the entire NYC metropolitan area has an estimated population of 19 million people.[4] We concentrate on accounts that posted more than 100 GPS-tagged tweets during the one-month data collection period. We refer to them as *geo-active users*. The social network of the 6,237 geo-active users is shown in Fig. 6.2.

In the remainder of this chapter, we review our method for automatic detection of Twitter messages that suggest the author contracted an infectious disease[5] (Sadilek *et al.*, 2012b). We then show how this information can be unified with geo, social, and temporal Twitter data to extract a strong signal of the impact of various previously elusive factors on human health. Finally, we develop a CRF model that leverages the labeled tweets and makes accurate predictions about people's health state.

---

[3] http://search.twitter.com/api/

[4] http://www.census.gov/popest/metro/

[5] In this work, such diseases include those with symptoms that overlap with, but are not necessarily limited to, influenza-like illness (http://en.wikipedia.org/wiki/Influenza-like_illness).

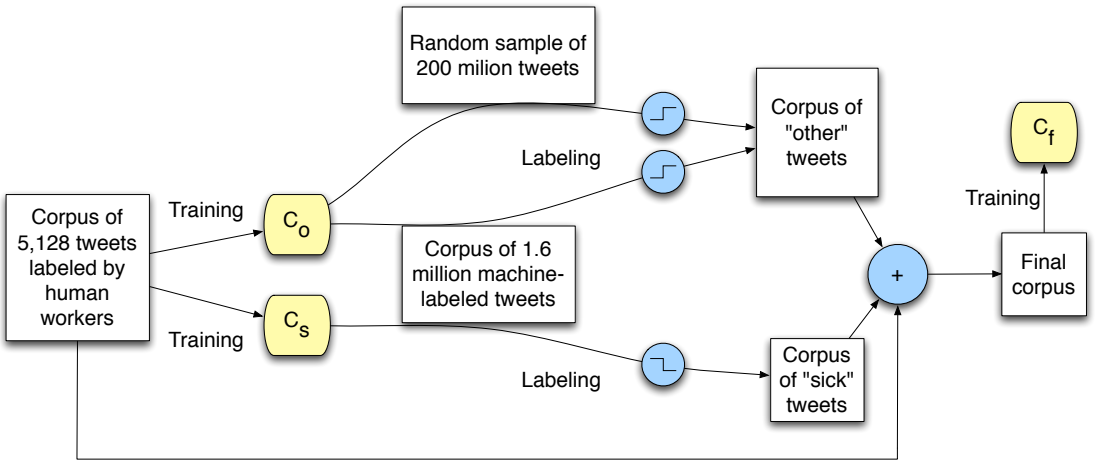


Figure 7.2: A diagram of our cascade learning of SVMs. The ⌐ and ¬ symbols denote thresholding of the classification score, where we select the bottom 10% of the scores predicted by C_o (*i.e.*, tweets that are normal with high probability), and the top 10% of scores predicted by C_s (*i.e.*, likely "sick" tweets).

7.3 Detecting Illness-Related Messages

As a first step, we need to identify Twitter messages that indicate the author is infected with a disease of interest at the time of posting. Based on the results of previous work, we expect that health-related tweets are relatively scarce as compared to other types of messages (Culotta, 2010; Paul and Dredze, 2011a). Given this class imbalance problem, we formulate a semi-supervised cascade-based approach (shown in Fig. 7.2) to learning a robust support vector machine (SVM) classifier with a large area under the ROC curve (*i.e.*, consistently high precision and high recall). SVM is an established model of data in machine learning (Cortes and Vapnik, 1995). We learn an SVM for linear binary classification to accurately distinguish between tweets indicating the author is afflicted by an infectious ailment (we call such tweets "sick"), and all other tweets (called "other" or "normal").

In order to learn such classifier, we ultimately need to effortlessly obtain a high-quality set of labeled training data. We achieve this via the following "bootstrapping" process. We begin by training two different binary SVM classifiers, C_s and C_o, using

the SVM^{light} package.[6] C_s is highly penalized for inducing false positives (mistakenly labeling a normal tweet as one about sickness), whereas C_o is heavily penalized for creating false negatives (labeling symptomatic tweets as normal). For both classifiers, the misclassification penalty for one direction was always a hundred times larger than in the opposite direction. We train C_s and C_o using a dataset of 5,128 tweets, each labeled as either "sick" or "other" by multiple Amazon Mechanical Turk workers and carefully checked by the authors. After training, we used the two classifiers to label a set of 1.6 million tweets that are likely health-related, but contain some noise. We obtained both datasets from (Paul and Dredze, 2011a), and they are completely disjoint from our NYC data.

The intuition behind this cascading process is to extract tweets that are with high confidence about sickness with C_s, and tweets that are almost certainly about other topics with C_o from the corpus of 1.6 million tweets. We further supplement the final corpus with messages from a sample of 200 million tweets (also disjoint from all other corpora considered here) that C_o classified as "other" with high probability. We apply thresholding on the classification score to reduce the noise in the cascade, as shown in Fig. 7.2.

The cascade yields a final corpus with over 700 thousand "sick" messages and 3 million "other" tweets, which we use for training the final SVM C_f. We will discuss how we leverage C_f to model the disease spread below, but first let us describe the feature space and our learning methodology in more detail.

As features, we use all unigram, bigram, and trigram word tokens that appear in the training data. For example, a tweet *"I feel sick."* is represented by the following feature vector:

$$\Big(i, feel, sick, i\ feel, feel\ sick, i\ feel\ sick\Big).$$

Before tokenization, we convert all text to lower case, strip punctuation and special characters, and remove mentions of user names (the "@" tag). All re-tweets (analogous to email forwarding) have been removed as well, since those messages typically refer

[6] http://svmlight.joachims.org/

to popular news and social games, and rarely describe the current state of the author. However, we do keep hashtags (such as "#sick"), as those are often relevant to the author's health state, and are particularly useful for disambiguation of short or ill-formed messages. When learning the final SVM C_f, we only consider tokens that appear at least three times in the training set.

While our feature space has a very high dimensionality (C_f operates in more than 1.7 million dimensions), with many possibly irrelevant features, support vector machines with a linear kernel have been shown to perform very well under such circumstances (Joachims, 2006; Sculley *et al.*, 2011; Paul and Dredze, 2011a).

To overcome the class imbalance problem, where the number of tweets about an illness is much smaller than the number of other messages, we apply the ROCArea SVM learning method that directly optimizes the area under the ROC curve, as described in (Joachims, 2005). Traditional objective functions, such as the 0-1 loss perform poorly under severe class imbalance. For instance, a trivial model that labels every example as belonging to the majority class has an excellent accuracy, because it misses only the relatively few minority examples. By contrast, the ROCArea method works by implicitly transforming the classical SVM learning problem over individual training examples into one over pairs of examples. This allows efficient calculation of the area under the ROC curve from the predicted ranking of the examples.

7.4 Patterns in the Spread of Disease

Human contact is the single most important factor in the transmission of infectious diseases (Clayton *et al.*, 1993). Since the contact is often indirect, such as via a doorknob, we focus on a more general notion of *co-location*. We consider two individuals co-located if they visit the same 100 by 100 meter cell within a time window (slack) of length T. For clarity, we show results for $T \in \{1, 4, 12\}$ hours. We use the 100 meter threshold, as that is the typical lower bound on the accuracy of a GPS sensor in obstructed areas, such as Manhattan. Since we focus on geo-active individuals, we can calculate co-location with high accuracy. The results below are for a condition, where a person is ill up to

two days after they write a "sick" tweet. It is important to note that the relationships among friendship, co-location, and health are consistent over a wide range of duration of contagiousness (from 1 to 7 days). Most infectious illnesses produce influenza-like symptoms that stop within a few days, and thus within these temporal bounds.

To quantify the effect of social ties on disease transmission, we leverage users' Twitter friendships. Clearly, there are complex events and interactions that take place "behind the scenes", which are not directly recorded in online social media. However, this work posits that these latent events often exhibit themselves in the activity of the sample of people we can observe. For instance, as we will see, having social ties to infected people significantly increases your chances of becoming ill in the near future. However, we do not believe that the social ties *themselves* cause or even facilitate the spread of an infection. Instead, the Twitter friendships are proxies and indicators for a complex set of phenomena that may not be directly accessible. For example, friends often eat out together, meet in classes, share items, and travel together. While most of these events are never explicitly mentioned online, they are crucial from the disease transmission perspective. However, their likelihood is modulated by the structure of the social ties, allowing us to reason about contagion.

Evaluation of the final SVM C_f described in the previous section on a held-out test set of 700,000 tweets shows 0.98 precision and 0.97 recall. This evaluation run also allows us to choose an optimal threshold on the classification score that separates the normal tweets from sick tweets. Table 7.2 lists the most significant features C_f found. Table 7.1 shows examples of tweets that C_f identified as "sick". We now apply C_f to modeling the spread of infectious diseases throughout the sampled population of NYC described above.

The correlation between the prevalence of infectious diseases predicted by our model and the predictions made by Google Flu Trends specifically for New York City is 0.73. The official CDC data for NYC is not available with sufficiently fine granularity, but previous work has shown that Google's predictions closely correspond to the official statistics for larger geographical areas (Ginsberg *et al.*, 2008). Google Flu Trends may have greater specificity to "influenza-like illness", whereas our approach may be less

Came home sick today from work with a killer headache and severe nausea, took 2 advil and slept for 6 hours. I feel much better now.
Meh I actually have to go to school tomorrrow.. #sick
Not feeling good at all...that sucks because I plans with my bff and job interviews set up until Tuesday. Stomach is killing me
I'm feeling better today still stuffed up but my nose isn't running like it was yesterday and my cough is better as well it hurts.
Guys I'm sorry. I'm really have to get some rest. I have nausea, headache, is tired, freezing & now have I got fever. Good Night! :-*
It hurts to breathe, swallow, cough or yawn. I must be getting sick, though because my ear feels worse than my throat.
I just sneezed 6 times in a row. i hate being sick.
feeling misserable. stomach hurts, headache, and no, I'm not pregnant.
Been sleep all day smh.... Currently soothing my jimmy frm α headache as I go back to sleep
Just not feeling it today. Looks like man flu has come back for a visit. I need to be well and have work - is that too much to ask?

Table 7.1: Example tweets that our SVM model C_f identified as "sick". Note the high degree of variability, and sometimes subtlety, in the way different people describe their health.

specific, but more sensitive to detect other, related infectious processes exhibiting these nonspecific features in Twitter content. Furthermore, the only overlap between our predictions and those of Google is for May 18 through 23, 2010. Thus, the correlation between the two needs to be interpreted with this context in mind.

Figures 7.3a and 7.3b show the impact of co-location and friendship with infected people on a given day on one's health the following day. We analyze both the individual and joint effects of the two factors on disease transmission. For brevity, we include plots only for a 1-day lag, since other time offsets result in a similar relationship.

Looking at co-location effect alone, we observe a definite exponential relationship between probable physical encounters and ensuing sickness. All three curves in Fig. 7.3a consistently fit $f(x) = C\ e^{(0.055x)}$, where C is a constant that captures the length of time overlap T (note that $C \simeq 0.011/T$; thus the larger the slack the smaller the effect). For instance, having 40 encounters with sick individuals with a 1-hour slack makes one ill with 20% probability. With a more lenient slack, such as 4 hours, one needs over 80 encounters to reach the same level of risk.

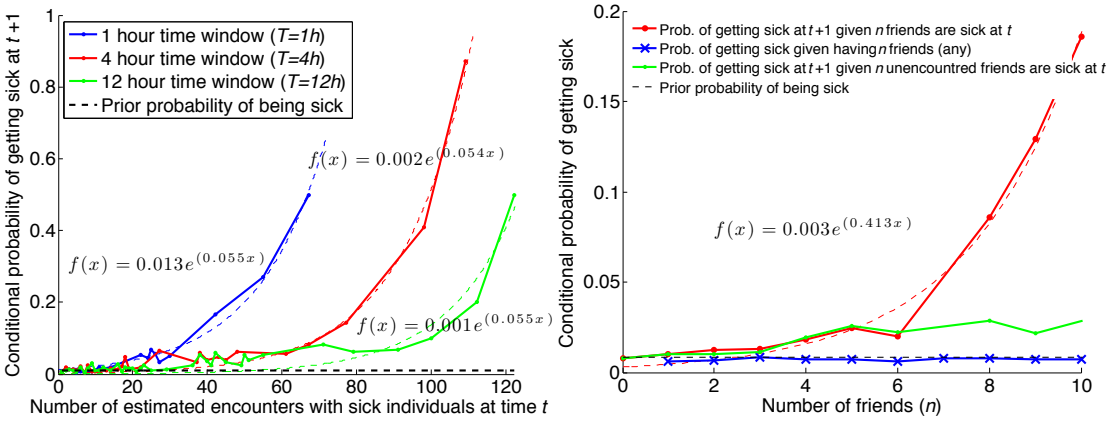
In Fig. 7.3b, we see that the number of sick friends also has an exponential effect on the probability of getting sick: $f(x) = 0.003\ e^{(0.413x)}$. By contrast, the number of friends in any health state (*i.e.*, the size of one's friend list) has *no* impact on one's health. In fact, the conditional probability of getting sick given n friends (the blue line in Fig. 7.3b) is virtually identical to the prior probability of getting sick (the black line).

We have discussed the latent influence of friendships earlier. This is quantitatively shown as the green line in Fig. 7.3b, where we *subtract out* the effect of co-location from the influence of social ties. We do this by counting only sick friends who have not been encountered. Comparison with the red curve shows that for smaller numbers of friends ($n \leq 6$), co-location has a weak additional effect over the proxy effect of social ties. However, for larger n, the residual impact of friendships plateaus, and co-location begins to dominate.

The ability to effortlessly quantify the sickness level for each individual in a large population enables us to gain insights into phenomena that were previously next to

Positive Features		**Negative Features**	
Feature	**Weight**	**Feature**	**Weight**
sick	0.9579	sick of	−0.4005
headache	0.5249	you	−0.3662
flu	0.5051	of	−0.3559
fever	0.3879	your	−0.3131
feel	0.3451	lol	−0.3017
cough	0.3062	who	−0.1816
feeling	0.3055	u	−0.1778
coughing	0.2917	love	−0.1753
throat	0.2842	it	−0.1627
cold	0.2825	her	−0.1618
home	0.2107	they	−0.1617
still	0.2101	people	−0.1548
bed	0.2088	shit	−0.1486
better	0.1988	smoking	−0.0980
being	0.1943	i'm sick of	−0.0894
being sick	0.1919	so sick of	−0.0887
stomach	0.1703	pressure	−0.0837
and my	0.1687	massage	−0.0726
infection	0.1686	i love	−0.0719
morning	0.1647	pregnant	−0.0639

Table 7.2: Top twenty most significant negatively and positively weighted features of our SVM model.



(a) Effects of co-location alone

(b) Effects of friendship and co-location

Figure 7.3: Being co-located with ill, symptomatic individuals, and having sick friends on a given day (t) makes one more likely to get sick the next day (t+1). On the horizontal axis in **(a)**, we plot the amount of co-location of an asymptomatic user with known sick people on a given day. In **(b)**, we show the number of friends (of an asymptomatic user); either only sick ones or any depending on the curve. The vertical axes show the conditional probability of getting sick the next day. We also plot the prior probability of being sick. For co-location, results for three slack time windows, within which we consider an appearance of two users close together as co-location, are shown (1, 4, and 12 hours).

impossible to capture. For instance, Fig. 7.4 shows the relationship between the health of New Yorkers inferred from social media by our system, and pollution sources in the city provided by the U.S. Environmental Protection Agency. We envision that unifications of rich datasets with inferred information in this fashion will empower individuals, organizations, as well as governments to make better informed data-driven decisions.

We can furthermore explore complex geographical and social patterns with fine-granularity and at a large scale as shown in Fig. 7.5. The United States has the world's largest health inequality across the social spectrum, where the gap of life expectancy of the most and the least advantaged segments of the population is over 20 years (Truman and others, 2011). It has been well-reported on that this difference is to a large part due to differences in social status. Our work is the first to enable the government as well as individuals to model the interplay between health, location, environment, and social factors effectively and with fine granularity.

Since we harness the sea of information in online social networks, we can infer and display the health of one's friends and people you encountered in a meaningful way. As a result, you can discover factors that impact your health and the health of people relevant to you. Once discovered, undesirable factors can be minimized while maximizing the positive variables.

In the remainder of this chapter, we show how the patterns revealed above can be leveraged in *fine-grained predictive* models of contagion.

7.5 Predicting the Spread of Disease

Human contact is the single most important factor in the transmission of infectious diseases (Clayton *et al.*, 1993). Since the contact is often indirect, such as via a doorknob, we focus on a more general notion of *co-location*. We consider two individuals co-located if they visit the same 100 by 100 meter cell within a time window (slack) of length T. For clarity, we show results for $T = 12$ hours, but we obtained virtually identical prediction performance for $T \in \{1, \ldots, 24\}$ hours. We use the $100m$ threshold, as that is the typical lower bound on the accuracy of a GPS sensor in obstructed areas, such as

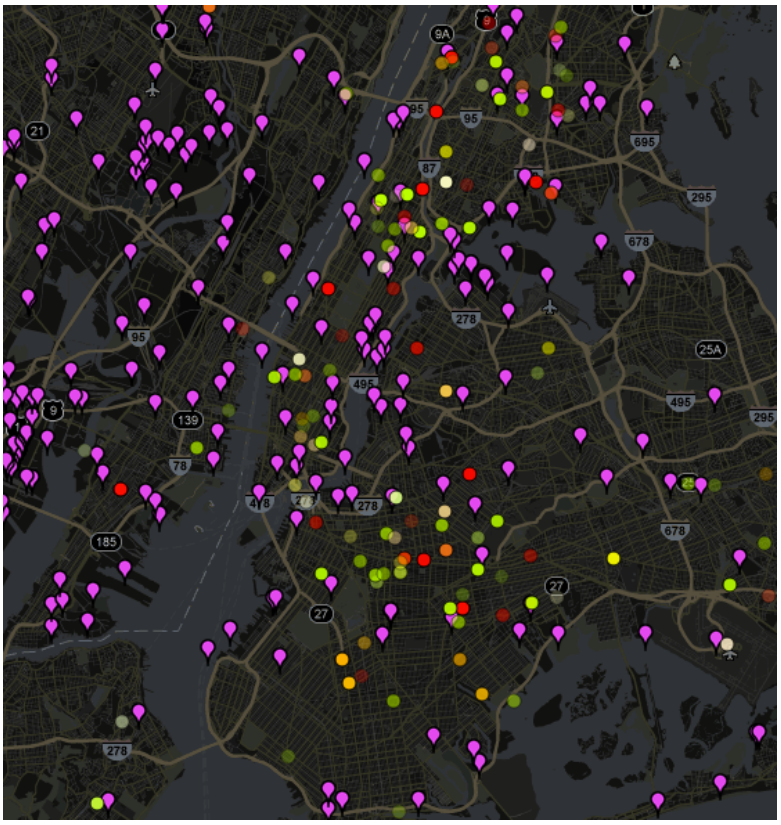


Figure 7.4: Visualization of the health and location of a sample of Twitter users (colored circles) and major pollution sources (purple pins) in New York City. Each circle shows a person's location on a map. Sick people are colored red, whereas healthy individuals are green. Since we are interested in pollution, by "sick" we mean people who exhibit upper respiratory tract symptoms, as inferred by our machine learning model. Time is displayed with opacity—the brightest markers have been infected most recently. Clicking on a pollution source displays information about the specific emissions it produces, and other information.

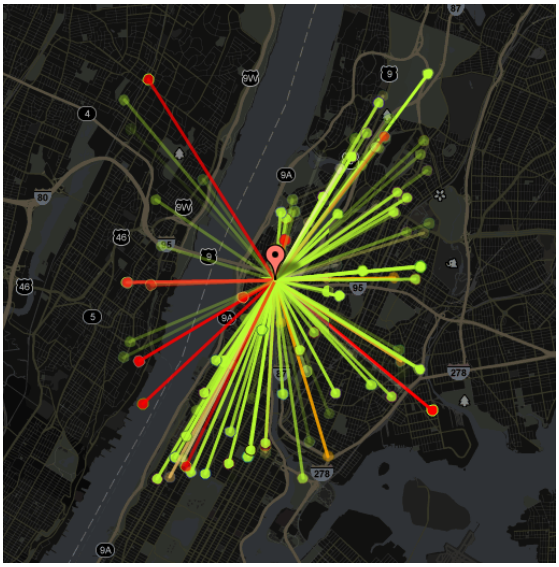


Figure 7.5: This figure shows the health status of people within a social network of the user at the center. Each lines represents a co-location relationship between two people, and the color denotes a person's health status. Note the patterns in geo-social distribution of sickness. For instance, people on the east side of the Hudson River tend to be more sick than nearby users from Manhattan.

Manhattan. Since we focus on geo-active individuals, we can calculate co-location with high accuracy. The results below are for a condition, where we consider a person ill up to four days after they write a "sick" tweet. As with the parameter T, it is important to note that the results are consistent over a wide range of duration of contagiousness (from 1 to 7 days). Few diseases with influenza-like symptoms are contagious for periods of time beyond these bounds.

Statistical analysis of the data shows that avoiding encounters with infected people generally decreases your chances of becoming ill, whereas a large amount of contact with them makes an onset of a disease almost certain (Sadilek *et al.*, 2012b). We find a definite exponential relationship between the intensity of co-location and the probability of getting ill. Similarly, by interpreting a Twitter friendship as a proxy for unobservable phenomena and interactions, we see that the likelihood of becoming ill increases as the number of infected friends grows. For example, having more than 5 sick friends increases one's likelihood of getting sick by a factor of 3, as compared to prior probability, and even more with respect to the probability given no sick friends. Additionally, we model the *joint* influence of co-location and social ties, and conclude that the latent impact of friendships is weaker (linear in the number of sick friends), but nonetheless important, as some observed patterns cannot be explained by co-location alone (Sadilek *et al.*, 2012b).

Our goal now is to leverage the interplay of co-location and friendships to predict the health state of any individual on a given day. For this purpose, we learn a dynamic conditional random field (CRF), a discriminative undirected graphical model (Lafferty, 2001). CRFs have been successfully applied in a wide range of domains from language understanding to robotics. They can systematically outperform alternative approaches, such as hidden Markov models, in domains where it is unrealistic to assume that observations are independent given the hidden state.

In our approach, each person X is captured by one dynamic CRF model with a linear chain structure shown in Fig. 7.6. Each time slice t contains one hidden binary random variable (X is either "healthy" or "sick" on day t), and a 25-element vector of

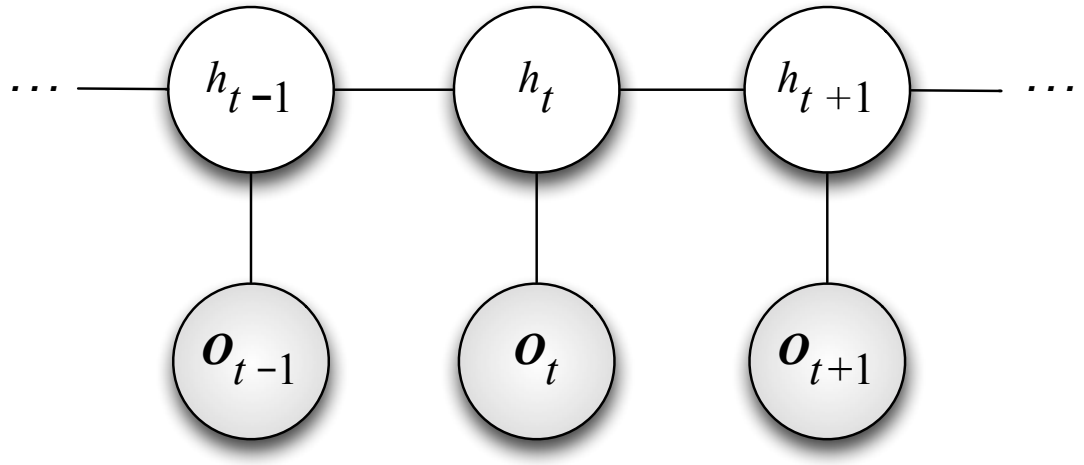


Figure 7.6: This conditional random field models the health of an individual over a number of days (h_t). The observations for each day ($\boldsymbol{o}_t$) include day of week, history of sick friends in the near past, the intensity of recent co-location with sick individuals, and the number of such individuals encountered.

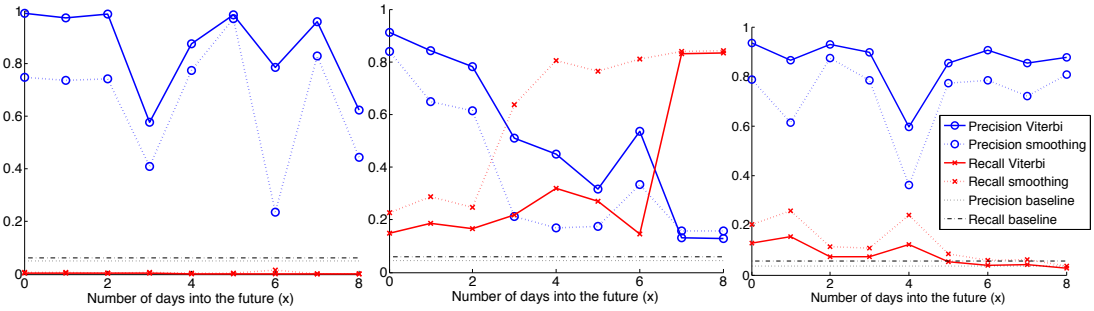
observed discrete random variables $\boldsymbol{o}_t$ given by

$$\boldsymbol{o}_t = \big(\text{weekday}, c_0, \ldots, c_7, u_0, \ldots, u_7, f_0, \ldots, f_7\big),$$

where c_n denotes the number of estimated encounters (co-locations) with sick individuals n days ago. For example, the value of c_1 indicates the number of co-location events a person had a day ago $(t-1)$, and c_0 shows co-location count for the current day t. Analogously, u_n and f_n denote the number of unique sick individuals encountered, and the number of sick Twitter friends, respectively, n days ago. For all random variables in our model, we use a special missing value to represent unavailable data.

7.5.1 Experiments and Results

In this section, we evaluate our approach in a number of experimental conditions, compare the results of our CRF models with a baseline, and discuss insights gained. We perform 6237-fold cross-validation (the number of geo-active users), where in order to make predictions for a given user, we train and test the CRF while treating all other users as observed. We report results aggregated over all cross-validation runs.



(a) Only Friends Observed (b) Only Co-Location Observed (c) Both Observed

Figure 7.7: Summary of results. Each plot shows the precision and recall of our three models for predictions made with hindsight ($x = 0$), and up to 8 days into the future ($x = 8$). We see that when leveraging the effect of social ties or co-locations individually (plots **(a)** and **(b)**, respectively) the CRF models perform inconsistently as we make predictions further into the future. By contrast, when considering friendships and co-location *jointly* **(c)**, the performance stabilizes and improves, achieving up to 0.94 precision and 0.18 recall (AUC of 0.85).

While the structure of the CRF model remains constant across our experiments, we consider two types of inference: Viterbi decoding, and the forwards-backwards algorithm (smoothing). While the former finds the most likely *sequence* of hidden variables (health states) given observations, the latter infers each state by finding maximal marginal probabilities. The tree structure of our CRF allows for scalable, yet exact, learning and inference by applying dynamic programming (Sutton and McCallum, 2006), while the rich temporal features capture longer-range dependencies in the data. L1 regularization is used to limit the number of parameters in our model. Maximum-likelihood parameter estimation is done via quasi-Newton method, and we are guaranteed to find a global optimum since the likelihood function is convex.

As a baseline, we consider a model that draws its predictions from a Bernoulli distribution with the "success" parameter p set to the prior probability of being sick learned from the training data.

Fig. 7.7 summarizes the performance of our models (Bernoulli baseline, and CRF

with Viterbi and forwards-backwards inference, respectively) in terms of precision and recall along two main dimensions. The first dimension is the type of features the CRF leverges: only information about sick friends (plus weekday) is observed in Fig. 7.7a; only co-location (plus weekday) is leveraged in Fig. 7.7b; and the full observation set $\boldsymbol{o}_t$ is available in Fig. 7.7c. The second dimension is the time for which we make predictions, shown on the horizontal axes (x) in Fig. 7.7. For $x = 0$, the plots show the performance when inferring the most likely health state for the entire observation sequence (*i.e.*, up to the present day). For $x > 0$, we show the precision and recall when predicting x days into the future, where observations are *not* available. (As described in the previous section, variables corresponding to future observations are simply set to the special "missing" value.)

We see that the results of our CRFs significantly outperform the baseline model. When leveraging the effect of social ties or co-locations individually (Figs. 7.7a and 7.7b, respectively), the CRF models perform inconsistently as we make predictions further into the future. By contrast, when considering friendships and co-location *jointly*, the performance stabilizes and improves, achieving up to 0.94 precision and 0.18 recall (Fig. 7.7c).

In general, we see that Viterbi decoding results in better precision and worse recall, whereas forwards-backwards inference yields slightly worse precision, but improves recall. The relatively low recall indicates that about 80% of infections occur without any evidence in social media as reflected in our features. For example, there are a number of instances of users getting ill even though they had no recent encounters with sick individuals and all their friends have been healthy for a long time.

Clearly, there are complex events and interactions that take place "behind the scenes", which are not directly recorded in online social media. However, this work posits that these latent events often exhibit themselves in the activity of the sample of people we can observe. For instance, we have seen that having online social ties to infected people significantly increases one's chances of becoming ill in the near future (Sadilek *et al.*, 2012b). However, we do not believe that the social ties *themselves* cause or even facilitate the spread of an infection. Instead, the Twitter friendships are proxies

and indicators for a complex set of phenomena that may not be directly accessible. For example, friends often eat out together, meet in classes, share items, and travel together. While most of these events are never explicitly mentioned online, they are crucial from the disease transmission perspective. However, their likelihood is modulated by the structure of the social ties, allowing us to reason about contagion.

7.6 Limitations

Our observations are limited by the prevalence of public tweets in which users talk about their health, and by our ability to identify them in the flood of other types of messages. Both these factors contribute to the fact that the number of infected individuals is systematically underestimated, but evaluation of C_f suggests that the latter effect is small. We can approximate the magnitude of this bias using the statistics presented earlier. We see that about 1 in 30 residents of NYC appears in our dataset. If we strictly focus on the geo-active individuals, the ratio is roughly 1:3,000. However, the results in this chapter indicate, that by leveraging the latent effects of our observations, such a sampling ratio is sufficient to predict the health state of a large fraction of the users with high precision.

We note that currently used methods suffer from similar biasing effects. For example, infected people who do not visit a doctor, or do not respond to surveys are virtually invisible to the traditional methods. Similarly, efforts such as Google Flu Trends can only observe individuals who search the web for certain types of content when sick. A fully comprehensive coverage of a population will require a combination of diverse methods, and application of AI techniques—like the ones presented in this work—capable of *inferring* the missing information.

7.7 Related Work

Since the famous cholera study by John Snow (1855), much work has been done in capturing the mechanisms of epidemics. There is ample previous work in computa-

tional epidemiology on building models of coarse-grained disease spread via differential equations (Anderson and May, 1979), by harnessing simulated populations (Eubank *et al.*, 2004), and by analysis of official statistics (Grenfell *et al.*, 2001). Such models are typically developed for the purposes of assessing the impact a particular combination of an outbreak and a vaccination or containment strategy would have on humanity, a country's defense, or ecology (Chen *et al.*, 2010). However, the above works focus on simulated populations and hypothetical scenarios. By contrast, we address the problem of assessing and modeling the health of *real-world* populations composed of individuals embedded in a fine social structure. As a result, our work is a major step towards prediction of actual threats and instances of disease outbreaks.

(Eubank *et al.*, 2004) are beginning to leverage more fine-grained information, including people's activities. They developed a simulation tool (EpiSims) that leverages synthetic—but statistically realistic—human mobility to study the spread of infectious diseases over a metropolitan area. They show their simulation-based approach is a viable alternative to the classical models formulated using differential equations. Crucially, Eubank *et al.* (2004) demonstrate that their methods enable accurate modeling a hypothetical spread of disease throughout a large, although in many ways artificial, population. This knowledge can in turn be used to seek an optimal emergency response policy.

In the context of social media, (Krieck *et al.*, 2011) explore augmenting the traditional notification channels about a disease outbreak with data extracted from Twitter. By manually examining a large number of tweets, they show that self-reported symptoms are the most reliable signal in detecting if a tweet is relevant to an outbreak or not. This is because people often do not know what their true problem is until diagnosed by an expert, but they can readily write about how they feel. Researchers have also concentrated on capturing the overall *trend* of a particular disease outbreak, typically influenza, by monitoring social media (Culotta, 2010; Lampos *et al.*, 2010; Chunara *et al.*, 2012). Interesting work of (Ritterman *et al.*, 2009) shows that noisy Twitter data is a valuable information channel for predicting public opinion regarding the likelihood of a pandemic. (Freifeld *et al.*, 2010) use information actively submitted by cell phone users

to model aggregate public health. However, scaling such systems poses considerable challenges.

Other researchers focus on a more detailed modeling of the *language* of the tweets and its relevance to public health in general (Paul and Dredze, 2011a), and to influenza surveillance in particular (Collier *et al.*, 2011). Paul et al. develop a variant of topic models that captures the symptoms and possible treatments for ailments, such traumatic injuries and allergies, that people discuss on Twitter. In a follow-up work (Paul and Dredze, 2011b) begin to consider the geographical patterns in the prevalence of such ailments, and show a good agreement of their models with official statistics and Google Flu Trends. There is a potential for synergy between the work of Paul et al. and ours that would allow us to model the spread of *specific* diseases by leveraging the rich language models.

However, all these works consider only aggregate patterns captured by coarse-grained statistics, whereas the primary contribution of our work is a more detailed study of the interplay among human mobility, social structure, and disease transmission. Our framework allows us to track—without active user participation—specific likely events of contagion between individuals, and model the relationship between an epidemic and self-reported symptoms of actual users of online social media.

Even the state of the art systems suffer from two major drawbacks. First, they produce only coarse, aggregate statistics, such as the expected number of people afflicted by flu in Texas. Furthermore, they often perform mere passive monitoring, and prediction is severely limited by the low resolution of the aggregate approach, or by scalability issues. By contrast, the primary contribution of our work is a fine-grained analysis of the interplay among human mobility, social structure, and disease transmission. Our framework allows us to make predictions about likely events of contagion between specific individuals without active user participation.

While we concentrate on "traditional" infectious diseases, such as flu, similar techniques can be applied to study mental health disorders, such as depression, that have strong contagion patterns as well. Pioneering work in this broad area includes (Silenzio *et al.*, 2009), which studies characteristics of young lesbian, gay, and bisexual individuals

in online social networks. They focus on discovering such members of a community, and design methods for effective peer-driven information diffusion and preventative care, focusing specifically on suicide. Twitter has also been used to monitor the seasonal variation in affect around the globe (Golder and Macy, 2011).

Looking at a more global scale, (Bettencourt and West, 2010) argue for a comprehensive scientific approach to urban planning. They show there are underlying patterns that tie together the size of a city with its emergent characteristics, such as crime rate, number of patents produced, walking speed of its inhabitants, and prevalence of epidemics. The authors argue that cities are the source of many major problems, but also contain the solutions because of their concentrated creativity and productivity.

Since this work leverages social ties and user location, the large body of prior work on inferring and predicting these characteristics becomes relevant. A number of researchers have demonstrated that it is possible to accurately predict people's fine-grained location from their online behavior and interactions (Cho *et al.*, 2011; Sadilek *et al.*, 2012a). Much progress has been made in predicting the social structure of participants in online media, including Twitter, from various types of observed data (Crandall *et al.*, 2010; Backstrom and Leskovec, 2011; Sadilek *et al.*, 2012a). Applying these machine learning techniques will significantly expand the breadth of data available by allowing us to consider not only declared friendships and public check-ins, but also their inferred—though more ambiguous—counterparts.

Finally, we note that the spread of disease is closely tied to the study of influence in social networks, where the usual goal is to identify the most influential members to facilitate information diffusion (Domingos and Richardson, 2001; Kempe *et al.*, 2003; Mossel and Roch, 2007). However, in conjunction with our previous work on location prediction and social network analysis, similar techniques can be applied to detect and treat key people in an epidemic. Both social structure and location of individuals—the two main focus areas of our unified model—have been shown to play major roles in epidemiology (Anderson and May, 1979; Keeling and Eames, 2005; Shakarian *et al.*, 2010), but have been studied only at an aggregate level.

7.8 Conclusions and Future Work

This work is the first to take on prediction of the spread of infectious diseases throughout a real-world population with fine granularity. We focus on self-reported symptoms that appear in people's Twitter status updates, and show that although such messages are rare, we can identify them with systematically high precision and high recall.

The key contribution of this work is a scalable probabilistic model that demonstrates that the health of a person can be accurately inferred from her location and social interactions observed via social media. Furthermore, we show that future health states can be *predicted* with consistently high accuracy more than a week into the future. For example, over 10% of cases of sickness are predicted with 90% confidence even a week before they occur. For predictions one day into the future, our model covers almost 20% of cases with the same confidence.

An early identification of infected individuals is especially crucial in preventing and containing devastating disease outbreaks. Important work by Eubank *et al.* (2004) shows that by far the most effective way to fight an epidemic in urban areas is to quickly confine infected individuals to their homes. However, this strategy is truly effective only when applied early on in the outbreak. The *speed* of targeted vaccination ranks second in effectiveness. This chapter shows that finding some of these key symptomatic individuals, along with other people that may have already contracted the disease, can be done effectively and in a timely manner through social media.

In future work, we will focus on larger geographical areas (including airplane travel), while maintaining the same level of detail (*i.e.*, social ties between concerete individuals and their fine-grained location). This will allow us to model and predict the emergence of global epidemics from the day-to-day interactions of individuals, and subsequently answer questions such as *"How did the current flu epidemic in city A start and where did it come from?"* and *"How likely I am to catch a cold if I visit the mall?"*

For example, Fig. 7.8 illustrates an instance in our dataset, where a sick person at an airport posts a message, and we can see other people nearby with whom he could have come into contact. Prior work has developed a repertoire of powerful AI techniques for

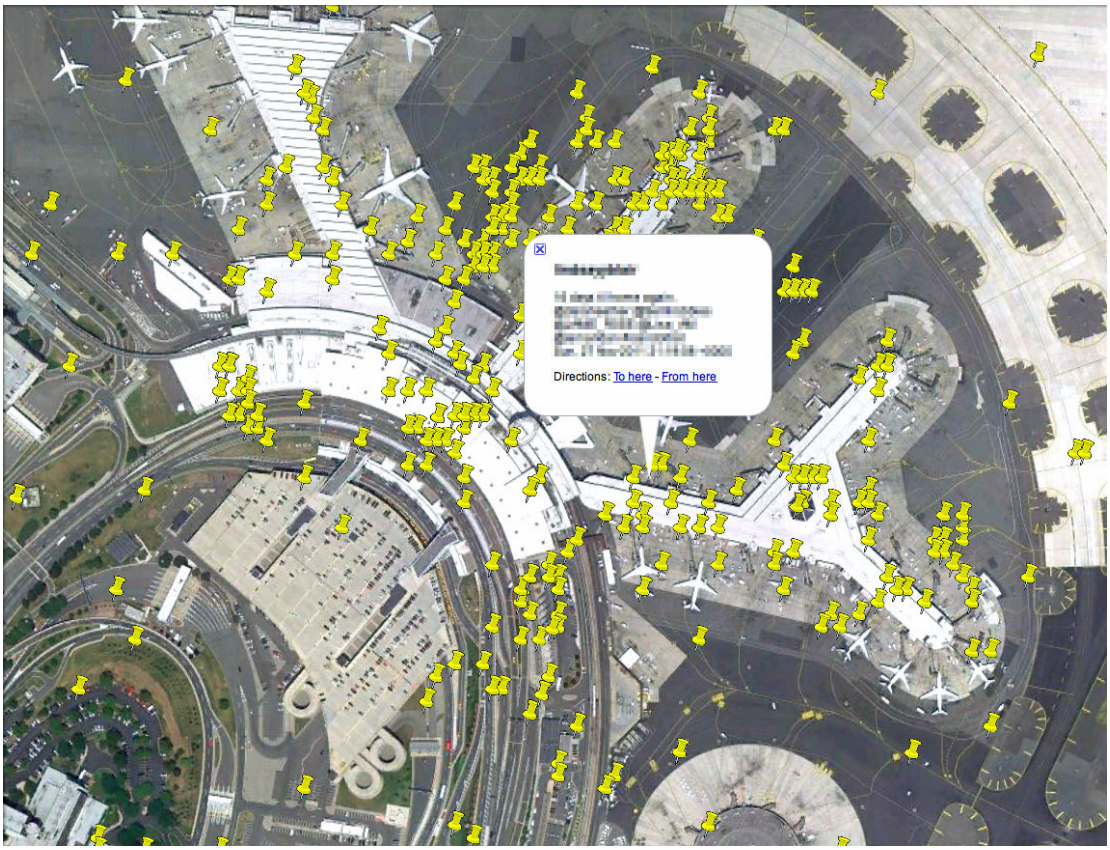


Figure 7.8: Visualization of a sample of Twitter users (yellow pins) at an airport. The highlighted person X says he will be back in 16 days and mentions specific friends for whom this message is relevant. We immediately see the people at the airport who could have come into contact with X. This work shows that we can accurately predict the health of X from his co-location with other individuals and the heath of his friends. However, additional information can be inferred using methods developed by previous work (Crandall *et al.*, 2010; Backstrom and Leskovec, 2011; Cho *et al.*, 2011; Sadilek *et al.*, 2012a). It can be expected that putting all this information together will yield even stronger and more comprehensive predictions about the spread of an infection.

revealing hidden social ties and predicting user location—two features heavily leveraged by our public health model. Therefore, there are opportunities for great synergy in these areas.

8 Conclusions

This thesis focuses on data mining of diverse, noisy, and incomplete sensory data over large numbers of individuals. We show that the mined patterns can be subsequently leveraged in predictive models of human activities, location, social interactions, and health at a large scale. We find that the raw sensory data linked with the content of users' online communication, the explicit as well as the implicit online social interactions, and relationships are extremely rich information sources. The fine granularity and pervasiveness of the data allows us to model phenomena that have been out of reach thus far. Furthermore, we can do so with a high degree of detail and at a population scale.

We recognize that there are substantial privacy questions ahead. For example, our work shows that much information can be *inferred* even for people who enabled privacy filters in their online profiles. We believe the privacy and "big brother" issues ultimately reduce to a cost-benefit analysis. Specifically, by quantifying the tradeoffs between the *value* our automated systems create versus loss of user privacy. In the future, we envision each person will be able to set a dollar valuation on his or her privacy, and online services will take that preference into account when collecting, analyzing, and using customer data. In one extreme, one may decide to risk sharing all data, make money on it, and get more personalized services. On the other hand, one may set tight privacy filters and pay for online services. This may lead to a new open marketplace with public data.

Bibliography

Gregory D. Abowd, Christopher G. Atkeson, Jason Hong, Sue Long, Rob Kooper, and Mike Pinkerton. Cyberguide: a mobile context-aware tour guide. *Wirel. Netw.*, 3(5):421–433, 1997.

E.L. Allwein, R.E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *The Journal of Machine Learning Research*, 1:113–141, 2001.

R.M. Anderson and R.M. May. Population biology of infectious diseases: Part I. *Nature*, 280(5721):361, 1979.

Daniel Ashbrook and Thad Starner. Using GPS to learn significant locations and predict movement across multiple users. *Personal Ubiquitous Comput.*, 7:275–286, October 2003.

S. Asur and B.A. Huberman. Predicting the future with social media. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on*, volume 1, pages 492–499. IEEE, 2010.

L. Backstrom and J. Leskovec. Supervised random walks: predicting and recommending links in social networks. In *WSDM 2011*, pages 635–644. ACM, 2011.

L. Backstrom, E. Sun, and C. Marlow. Find me if you can: improving geographical prediction with social and spatial proximity. In *Proceedings of the 19th international conference on World wide web*, pages 61–70. ACM, 2010.

C. Baker, J. Tenenbaum, and R. Saxe. Bayesian models of human action understanding. *Advances in Neural Information Processing Systems*, 18:99, 2006.

C.L. Baker, J.B. Tenenbaum, and R.R. Saxe. Goal inference as inverse planning. In *Proceedings of the 29th annual meeting of the cognitive science society*, 2007.

C.L. Baker, N.D. Goodman, and J.B. Tenenbaum. Theory-based social goal inference. In *Proceedings of the thirtieth annual conference of the cognitive science society*, pages 1447–1452, 2008.

C.L. Baker, R.R. Saxe, and J.B. Tenenbaum. Bayesian theory of mind: Modeling joint belief-desire attribution. In *Proceedings of the Thirty-Second Annual Conference of the Cognitive Science Society*, 2011.

Dare A. Baldwin and Jodie A. Baird. Discerning intentions in dynamic human action. *Trends in Cognitive Sciences*, 5(4):171 – 178, 2001.

Shirli Bar-David, Israel Bar-David, Paul C. Cross, Sadie J. Ryan, Christiane U. Knechtel, and Wayne M. Getz. Methods for assessing movement path recursion with application to african buffalo in south africa. In *Ecology*, 2009.

M. Barbuceanu and M.S. Fox. COOL: a language for describing coordination in multi agent systems. In *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, pages 17–24, 1995.

MG Beiró, JI Alvarez-Hamelin, and JR Busch. A low complexity visualization tool that helps to perform complex systems analysis. *New Journal of Physics*, 10:125003, 2008.

Robert Bell, Yehuda Koren, and Chris Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *KDD*, pages 95–104, New York, NY, USA, 2007. ACM.

A. Ben-Israel and T.N.E. Greville. *Generalized inverses: theory and applications*, volume 15. Springer Verlag, 2003.

C.M. Bergman, J.A. Schaefer, and SN Luttich. Caribou movement as a correlated random walk. *Oecologia*, 123(3):364–374, 2000.

L. Bettencourt and G. West. A unified theory of urban living. *Nature*, 467(7318):912–913, 2010.

M. Biba, S. Ferilli, and F. Esposito. Discriminative structure learning of Markov logic networks. pages 59–76. Springer, 2008.

Rahul Biswas, Sebastian Thrun, and Kikuo Fujimura. Recognizing activities with multiple cues. In *Workshop on Human Motion*, pages 255–270, 2007.

J. Blackburn, R. Simha, N. Kourtellis, X. Zuo, C. Long, M. Ripeanu, J. Skvoretz, and A. Iamnitchi. Cheaters in the steam community gaming social network. *Arxiv preprint arXiv:1112.4915*, 2011.

Leo Breiman et al. *Classification and Regression Trees*. Chapman & Hall, New York, 1984.

E.O. Brigham and RE Morrow. The fast Fourier transform. *Spectrum, IEEE*, 4(12):63–70, 1967.

A.J. Bernheim Brush, John Krumm, and James Scott. Exploring end user preferences for location obfuscation, location-based services, and the value of location. In *Ubicomp*, Ubicomp '10, pages 95–104, New York, NY, USA, 2010. ACM.

H. H. Bui. A general model for online probabilistic plan recognition. In *Eighteenth International Joint Conference on Artificial Intelligence (IJCAI-2003)*, 2003.

P. Busetta, L. Serafini, D. Singh, and F. Zini. Extending multi-agent cooperation by overhearing. In *Cooperative Information Systems*, pages 40–52. Springer, 2001.

H. Cao, N. Mamoulis, and D.W. Cheung. Discovery of periodic patterns in spatiotemporal sequences. *IEEE transactions on knowledge and data engineering*, pages 453–467, 2007.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41:15:1–15:58, July 2009.

P.A. Chen, M. David, and D. Kempe. Better vaccination strategies for better people. In *Proceedings of the 11th ACM conference on Electronic commerce*, pages 179–188. ACM, 2010.

K.H. Chiang and N. Shenoy. A 2-d random-walk mobility model for location-management studies in wireless networks. *Vehicular Technology, IEEE Transactions on*, 53(2):413–424, 2004.

E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: User movement in location-based social networks. *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2011.

R. Chunara, J.R. Andrews, and J.S. Brownstein. Social and news media enable estimation of epidemiological patterns early in the 2010 Haitian cholera outbreak. *The American Journal of Tropical Medicine and Hygiene*, 86(1):39–45, 2012.

K.C. Clarke and L.J. Gaydos. Loose-coupling a cellular automaton model and gis: long-term urban growth prediction for san francisco and washington/baltimore. *International Journal of Geographical Information Science*, 12(7):699–714, 1998.

D. Clayton, M. Hills, and A. Pickles. *Statistical models in epidemiology*, volume 41. Oxford university press Oxford, 1993.

N. Collier, N.T. Son, and N.M. Nguyen. OMG U got flu? Analysis of shared health messages for bio-surveillance. *Journal of Biomedical Semantics*, 2011.

C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

D.J. Crandall, L. Backstrom, D. Cosley, S. Suri, D. Huttenlocher, and J. Kleinberg. Inferring social ties from geographic coincidences. *Proceedings of the National Academy of Sciences*, 107(52):22436, 2010.

K. Cui, Z. Cao, X. Zheng, D. Zeng, K. Zeng, and M. Zheng. A geospatial analysis on the potential value of news comments in infectious disease surveillance. *Intelligence and Security Informatics*, pages 85–93, 2011.

A. Culotta and A. McCallum. Joint deduplication of multiple record types in relational data. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 257–258. ACM, 2005.

A. Culotta. Towards detecting influenza epidemics by analyzing Twitter messages. In *Proceedings of the First Workshop on Social Media Analytics*, pages 115–122. ACM, 2010.

M. Datar, N. Immorlica, P. Indyk, and V.S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry*, pages 253–262. ACM, 2004.

J. Davis and P. Domingos. Deep transfer via second-order Markov logic. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 217–224. ACM, 2009.

L. De Raedt and Kristian Kersting. Probabilistic inductive logic programming. (De Raedt *et al.*, 2008), pages 1–27.

L. De Raedt, Paolo Frasconi, Kristian Kersting, and Stephen Muggleton, editors. *Probabilistic Inductive Logic Programming - Theory and Applications*, volume 4911 of *Lecture Notes in Computer Science*. Springer, 2008.

L. De Raedt. *Logical and relational learning*. Springer-Verlag New York Inc, 2008.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

P. Denis and J. Baldridge. Joint determination of anaphoricity and coreference resolution using integer programming. In *Proceedings of NAACL HLT*, pages 236–243, 2007.

P. Domingos and M. Richardson. Mining the network value of customers. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 57–66. ACM, 2001.

Pedro Domingos, Stanley Kok, Daniel Lowd, Hoifung Poon, Matthew Richardson, and Parag Singla. Markov logic. (De Raedt *et al.*, 2008), pages 92–117.

P. Domingos. Multi-relational record linkage. In *In Proceedings of the KDD-2004 Workshop on Multi-Relational Data Mining*, 2004.

N. Eagle and A. Pentland. Reality mining: sensing complex social systems. *Personal and Ubiquitous Computing*, 10(4):255–268, 2006.

N. Eagle and A.S. Pentland. Eigenbehaviors: Identifying structure in routine. *Behavioral Ecology and Sociobiology*, 63(7):1057–1066, 2009.

Nathan Eagle, Alex Pentland, and David Lazer. Inferring social network structure using mobile phone data. In *Proceedings of the National Academy of Sciences*, 2009.

D. Easley and J. Kleinberg. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, 2010.

F. Ekman, A. Keränen, J. Karvo, and J. Ott. Working day movement model. In *Proceeding of the 1st ACM SIGMOBILE workshop on Mobility models*, pages 33–40. ACM, 2008.

S. Eubank, H. Guclu, VS Anil Kumar, M.V. Marathe, A. Srinivasan, Z. Toroczkai, and N. Wang. Modelling disease outbreaks in realistic urban social networks. *Nature*, 429(6988):180–184, 2004.

C.C. Freifeld, R. Chunara, S.R. Mekaru, E.H. Chan, T. Kass-Hout, A.A. Iacucci, and J.S. Brownstein. Participatory epidemiology: use of mobile phones for community-based health reporting. *PLoS medicine*, 7(12):e1000376, 2010.

N. Friedman, L. Getoor, D. Koller, and A. Pfeffer. Learning probabilistic relational models. In *International Joint Conference on Artificial Intelligence*, volume 16, pages 1300–1309, 1999.

P. Fulop, S. Szabo, and T. Szálka. Accuracy of random walk and Markovian mobility models in location prediction methods. In *Software, Telecommunications and Com-*

puter Networks, 2007. SoftCOM 2007. 15th International Conference on, pages 1–5. IEEE, 2007.

Z. Ghahramani. Learning dynamic Bayesian networks. In *Adaptive Processing of Sequences and Data Structures*, page 168. Springer, 1998.

J. Ginsberg, M.H. Mohebbi, R.S. Patel, L. Brammer, M.S. Smolinski, and L. Brilliant. Detecting influenza epidemics using search engine query data. *Nature*, 457(7232):1012–1014, 2008.

S.A. Golder and M.W. Macy. Diurnal and seasonal mood vary with work, sleep, and daylength across diverse cultures. *Science*, 333(6051):1878–1881, 2011.

M.C. González, C.A. Hidalgo, and A.L. Barabási. Understanding individual human mobility patterns. *Nature*, 453(7196):779–782, 2008.

C. Goutte and E. Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. pages 345–359. Springer, 2005.

BT Grenfell, ON Bjornstad, and J. Kappey. Travelling waves and spatial hierarchies in measles epidemics. *Nature*, 414(6865):716–723, 2001.

Anatoliy Gruzd, Barry Wellman, and Yuri Takhteyev. Imagining Twitter as an imagined community. In *American Behavioral Scientist, Special issue on Imagined Communities*, 2011.

Abhinav Gupta, Praveen Srinivasan, Jianbo Shi, and Larry S. Davis. Understanding videos, constructing plots: Learning a visually grounded storyline model from annotated videos. In *CVPR*, 2009.

B. Gutmann and K. Kersting. TildeCRF: conditional random fields for logical sequences. In *Machine Learning: ECML 2006*, pages 174–185. Springer, 2006.

Q. Hao, R. Cai, C. Wang, R. Xiao, J.M. Yang, Y. Pang, and L. Zhang. Equip tourists with knowledge mined from travelogues. In *Proceedings of the 19th international conference on World wide web*, pages 401–410. ACM, 2010.

R. Helaoui, M. Niepert, and H. Stuckenschmidt. A statistical-relational activity recognition framework for ambient assisted living systems. In *Ambient Intelligence and Future Trends-International Symposium on Ambient Intelligence (ISAmI 2010)*, pages 247–254. Springer, 2010.

Lichan Hong, Bongwon Suh, and Ed H. Chi. Tweets from justin bieber's heart: the dynamics of the "location" field in user profiles. In *ACM CHI*, 2011.

Jun Hong. Goal recognition through goal graph analysis. *Journal of Artificial Intelligence Research*, 15:1–30, 2001.

E. Horvitz, J. Apacible, R. Sarin, and L. Liao. Prediction, expectation, and surprise: Methods, designs, and study of a deployed traffic forecasting service. In *Twenty-First Conference on Uncertainty in Artificial Intelligence*, 2005.

W. Hsu, D. Dutta, and A. Helmy. Mining behavioral groups in large wireless lans. In *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 338–341. ACM, 2007.

W. Hsu, T. Spyropoulos, K. Psounis, and A. Helmy. Modeling time-variant user mobility in wireless mobile networks. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 758–766. IEEE, 2007.

D.H. Hu, S.J. Pan, V.W. Zheng, N.N. Liu, and Q. Yang. Real world activity recognition with multiple goals. In *UbiComp*, volume 8, pages 30–39, 2008.

T.N. Huynh and R.J. Mooney. Discriminative structure and parameter learning for Markov logic networks. In *Proceedings of the 25th international conference on Machine learning*, pages 416–423. ACM, 2008.

M. Jaeger. Relational Bayesian networks. In *Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence*, pages 266–273, 1997.

A. Jardosh, E.M. Belding-Royer, K.C. Almeroth, and S. Suri. Towards realistic mobility models for mobile ad hoc networks. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 217–229. ACM, 2003.

Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. Why we Twitter: understanding microblogging usage and communities. In *WebKDD/SNA-KDD*, pages 56–65, New York, NY, USA, 2007. ACM.

H. Jeung, Q. Liu, H.T. Shen, and X. Zhou. A hybrid prediction model for moving objects. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 70–79. IEEE, 2008.

S. Jiang, D. He, and J. Rao. A prediction-based link availability estimation for mobile ad hoc networks. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1745–1752. IEEE, 2001.

B. Jiang, J. Yin, and S. Zhao. Characterizing the human mobility pattern in a large street network. *Physical Review E*, 80(2):021136, 2009.

T. Joachims. A support vector method for multivariate performance measures. In *ICML 2005*, pages 377–384. ACM, 2005.

T. Joachims. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 217–226. ACM, 2006.

I. Jolliffe. Principal component analysis. *Encyclopedia of Statistics in Behavioral Science*, 2002.

M.I. Jordan. *Learning in graphical models*. Kluwer Academic Publishers, 1998.

E. Kamar and E. Horvitz. Collaboration and shared plans in the open world: Studies of ridesharing. In *IJCAI*, 2009.

Gal A. Kaminka, David V. Pynadath Milind Tambe, David V. Pynadath, and Milind Tambe. Monitoring teams by overhearing: A multi-agent plan-recognition approach. *Journal of Artificial Intelligence Research*, 17:2002, 2002.

M.J. Keeling and K.T.D. Eames. Networks and epidemic models. *Journal of the Royal Society Interface*, 2(4):295–307, 2005.

D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

K. Kersting and L. De Raedt. Bayesian logic programs. In *Proceedings of the Work-in-Progress Track at the 10th International Conference on Inductive Logic Programming*, 2000.

K. Kersting, L. De Raedt, and T. Raiko. Logical hidden Markov models. *Journal of Artificial Intelligence Research*, 25(1):425–456, 2006.

M. Kim and D. Kotz. Periodic properties of user mobility and access-point popularity. *Personal and Ubiquitous Computing*, 11(6):465–479, 2007.

M. Kim and D. Kotz. Identifying unusual days. *Journal of Computing Science and Engineering*, 5(1):71–84, 2011.

M. Kim, D. Kotz, and S. Kim. Extracting a mobility model from real user traces. In *Proc. IEEE Infocom*, pages 1–13. Citeseer, 2006.

S. Kok and P. Domingos. Learning the structure of Markov logic networks. In *Proceedings of the 22nd international conference on Machine learning*, pages 441–448. ACM, 2005.

S. Kok and P. Domingos. Statistical predicate invention. In *Proceedings of the 24th international conference on Machine learning*, pages 433–440. ACM, 2007.

Stanley Kok and Pedro Domingos. Statistical predicate invention. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 433–440, New York, NY, USA, 2007. ACM.

S. Kok and P. Domingos. Learning Markov logic network structure via hypergraph lifting. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 505–512. ACM, 2009.

D. Koller. Probabilistic relational models. In *Inductive Logic Programming*, pages 3–13. Springer, 1999.

M. Krieck, J. Dreesman, L. Otrusina, and K. Denecke. A new age of public health: Identifying disease outbreaks by analyzing tweets. *Proceedings of Health WebScience Workshop, ACM Web Science Conference*, 2011.

J. Krumm and A. Brush. Learning time-based presence probabilities. *Pervasive Computing*, pages 79–96, 2011.

J. Krumm and E. Horvitz. Predestination: Inferring destinations from partial trajectories. *UbiComp 2006: Ubiquitous Computing*, pages 243–260, 2006.

Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is Twitter, a Social Network or a News Media? In *WWW*, April 2010.

John Lafferty. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning (ICML)*, pages 282–289. Morgan Kaufmann, 2001.

V. Lampos, T. De Bie, and N. Cristianini. Flu detector-tracking epidemics on Twitter. *Machine Learning and Knowledge Discovery in Databases*, pages 599–602, 2010.

N. Landwehr, B. Gutmann, I. Thon, M. Philipose, and L. De Raedt. Relational transformation-based tagging for human activity recognition. In *Proceedings of the 6th International Workshop on Multi-relational Data Mining (MRDM07)*, pages 81–92, 2007.

R. Lee and K. Sumiya. Measuring geographical regularities of crowd behaviors for Twitter-based geo-social event detection. In *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks*, pages 1–10. ACM, 2010.

K. Lee, S. Hong, S.J. Kim, I. Rhee, and S. Chong. Slaw: A new mobility model for human walks. In *INFOCOM 2009, IEEE*, pages 855–863. IEEE, 2009.

Z. Li, B. Ding, J. Han, R. Kays, and P. Nye. Mining periodic behaviors for moving objects. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1099–1108. ACM, 2010.

Lin Liao, Dieter Fox, and Henry Kautz. Learning and inferring transportation routines. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence*, 2004.

Lin Liao, Dieter Fox, and Henry Kautz. Location-based activity recognition using relational Markov networks. In *IJCAI*, 2005.

L. Liao, D. Fox, and H. Kautz. Extracting places and activities from gps traces using hierarchical conditional random fields. *International Journal of Robotics Research*, 2007.

L. Liao, D.J. Patterson, D. Fox, and H. Kautz. Learning and inferring transportation routines. *Artificial Intelligence*, 171(5-6):311–331, 2007.

David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *J. Am. Soc. Inf. Sci. Technol.*, 58:1019–1031, May 2007.

D. Liben-Nowell, J. Novak, R. Kumar, P. Raghavan, and A. Tomkins. Geographic routing in social networks. *Proceedings of the National Academy of Sciences of the United States of America*, 102(33):11623, 2005.

B. Limketkai, D. Fox, and Lin Liao. CRF-filters: Discriminative particle filters for sequential state estimation. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 3142–3147, 2007.

X. Ling and D.S. Weld. Temporal information extraction. In *Proceedings of the Twenty Fifth National Conference on Artificial Intelligence*, 2010.

X. Liu and H.A. Karimi. Location awareness through trajectory prediction. *Computers, Environment and Urban Systems*, 30(6):741–756, 2006.

T. Liu, P. Bahl, and I. Chlamtac. Mobility modeling, location tracking, and trajectory prediction in wireless atm networks. *Selected Areas in Communications, IEEE Journal on*, 16(6):922–936, 1998.

Z. Ma. *Modelling with PRISM of intelligent system.* MSc. Thesis, Linacre College, University of Oxford, 2008.

C. Manfredotti and E. Messina. Relational dynamic Bayesian networks to improve multi-target tracking. In *Advanced Concepts for Intelligent Vision Systems*, pages 528–539. Springer, 2009.

Cristina Manfredotti, Howard Hamilton, and Sandra Zilles. Learning RDBNs for activity recognition. In *Neural Information Processing Systems*, 2010.

C. Manfredotti. Modeling and inference with relational dynamic Bayesian networks. In *Advances in Artificial Intelligence*, pages 287–290. Springer, 2009.

L. Mihalkova and R.J. Mooney. Bottom-up learning of Markov logic network structure. In *Proceedings of the 24th international conference on Machine learning*, pages 625–632. ACM, 2007.

S. Moon and A. Helmy. Understanding periodicity and regularity of nodal encounters in mobile networks: A spectral analysis. *Arxiv preprint arXiv:1004.4437*, 2010.

Darnell Moore and Irfan Essa. Recognizing multitasked activities using stochastic context-free grammar. In *In Proceedings of AAAI Conference*, 2001.

E. Mossel and S. Roch. On the submodularity of influence in social networks. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 128–134. ACM, 2007.

S. Muggleton. Learning structure and parameters of stochastic logic programs. In *Proceedings of the 12th international conference on Inductive logic programming*, pages 198–206. Springer-Verlag, 2002.

Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *In Proceedings of Uncertainty in AI*, pages 467–475, 1999.

Kevin P. Murphy. *Dynamic bayesian networks: representation, inference and learning*. PhD thesis, University of California, Berkeley, 2002.

M. Musolesi and C. Mascolo. Mobility models for systems evaluation. a survey, 2009.

Mor Naaman, Jeffrey Boase, and Chih-Hui Lai. Is it really about me?: message content in social awareness streams. In *CSCW '10: Proceedings of the 2010 ACM conference on Computer supported cooperative work*, pages 189–192, New York, NY, USA, 2010. ACM.

S. Natarajan, P. Tadepalli, E. Altendorf, T.G. Dietterich, A. Fern, and A. Restificar. Learning first-order probabilistic models with combining rules. In *Proceedings of the 22nd international conference on Machine learning*, pages 609–616. ACM, 2005.

Sriraam Natarajan, Hung H. Bui, Prasad Tadepalli, Kristian Kersting, and Weng-Keen Wong. Logical hierarchical hidden Markov models for modeling user activities. In *In Proc. of ILP-08*, 2008.

A. Noulas, S. Scellato, R. Lambiotte, M. Pontil, and C. Mascolo. A tale of many cities: universal patterns in human urban mobility. *Arxiv preprint arXiv:1108.5355*, 2011.

T. Papai, P. Singla, and H. Kautz. Constraint propagation for efficient inference in Markov logic. In *Seventeenth International Conference on Principles and Practice of Constraint Programming*, 2011.

M.J. Paul and M. Dredze. A model for mining public health topics from Twitter. *Technical Report. Johns Hopkins University. 2011.*, 2011.

M.J. Paul and M. Dredze. You are what you tweet: Analyzing Twitter for public health. In *Fifth International AAAI Conference on Weblogs and Social Media*, 2011.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

R. Penrose. On best approximate solutions of linear matrix equations. In *Proceedings of the Cambridge Philosophical Society*, volume 52, pages 17–19. Cambridge Univ Press, 1956.

Alex (Sandy) Pentland. *Honest Signals: How They Shape Our World*. The MIT Press, 2008.

H. Poon and P. Domingos. Sound and efficient inference with probabilistic and deterministic dependencies. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, page 458. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.

H. Poon and P. Domingos. Joint inference in information extraction. In *Proceedings of the 22nd national conference on Artificial intelligence-Volume 1*, pages 913–918. AAAI Press, 2007.

H. Poon and P. Domingos. Joint unsupervised coreference resolution with Markov logic. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 650–659. Association for Computational Linguistics, 2008.

F. Ramos, D. Fox, and H. Durrant-Whyte. Crf-matching: Conditional random fields for feature-based scan matching. In *Proc. of Robotics: Science and Systems*, 2007.

I. Rhee, M. Shin, S. Hong, K. Lee, S.J. Kim, and S. Chong. On the levy-walk nature of human mobility. *Networking, IEEE/ACM Transactions on*, 19(3):630–643, 2011.

Sebastian Riedel. Improving the accuracy and efficiency of map inference for Markov logic. In *Proceedings of the Proceedings of the Twenty-Fourth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-08)*, pages 468–475, Corvallis, Oregon, 2008. AUAI Press.

J. Ritterman, M. Osborne, and E. Klein. Using prediction markets and Twitter to predict a swine flu pandemic. *1st International Workshop on Mining Social Media*, 2009.

B. Roark, M. Saraclar, M. Collins, and M. Johnson. Discriminative language modeling with conditional random fields and the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 47. Association for Computational Linguistics, 2004.

Adam Sadilek and Henry Kautz. Modeling and reasoning about success, failure, and

intent of multi-agent activities. In *Mobile Context-Awareness Workshop, Twelfth ACM International Conference on Ubiquitous Computing*, 2010.

Adam Sadilek and Henry Kautz. Recognizing multi-agent activities from GPS data. In *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

Adam Sadilek and Henry Kautz. Location-based reasoning about complex multi-agent behavior. *Journal of Artificial Intelligence Research*, 43:87–133, 2012.

Adam Sadilek and Henry Kautz. Modeling success, failure, and intent of multi-agent activities under severe noise. *Mobile Context Awareness*, pages 9–63, 2012.

Adam Sadilek and John Krumm. Far out: Predicting long-term human mobility. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

Adam Sadilek, Henry Kautz, and Jeffrey P. Bigham. Finding your friends and following them to where you are. In *Fifth ACM International Conference on Web Search and Data Mining*, 2012. (Best Paper Award).

Adam Sadilek, Henry Kautz, and Vincent Silenzio. Modeling spread of disease from social interactions. In *Sixth AAAI International Conference on Weblogs and Social Media (ICWSM)*, 2012.

Adam Sadilek, Henry Kautz, and Vincent Silenzio. Predicting disease transmission from geo-tagged micro-blog data. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

T. Sato and Y. Kameya. Parameter learning of logic programs for symbolic-statistical modeling. In *Journal of Artificial Intelligence Research*, 2001.

T. Sato and Y. Kameya. New advances in logic-based probabilistic modeling by PRISM. In *Probabilistic inductive logic programming*, pages 118–155. Springer, 2008.

S. Scellato, M. Musolesi, C. Mascolo, V. Latora, and A. Campbell. Nextplace: A spatio-temporal prediction framework for pervasive systems. *Pervasive Computing*, pages 152–169, 2011.

S. Scellato, A. Noulas, R. Lambiotte, and C. Mascolo. Socio-spatial properties of online location-based social networks. *Proceedings of ICWSM*, 11, 2011.

J. Scott, A.J.B. Brush, J. Krumm, B. Meyers, M. Hazas, S. Hodges, and N. Villar. Preheat: controlling home heating using occupancy prediction. In *Proceedings of the 13th international conference on Ubiquitous computing*, pages 281–290. ACM, 2011.

D. Sculley, M. Otey, M. Pohl, B. Spitznagel, J. Hainsworth, and Z. Yunkai. Detecting adversarial advertisements in the wild. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011.

F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 134–141. Association for Computational Linguistics, 2003.

Paulo Shakarian, V.S. Subrahmanian, and Maria Luisa Spaino. SCARE: A Case Study with Baghdad. In *Proceedings of the Third International Conference on Computational Cultural Dynamics*. AAAI, 2009.

P. Shakarian, VS Subrahmanian, M.L. Sapino, M. Hermenegildo, and T. Schaub. Using generalized annotated programs to solve social network optimization problems. *ICLP*, 2010.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *Proceedings of the 24th international conference on Machine learning*, pages 807–814. ACM, 2007.

J. Shen. *Activity recognition in desktop environments*. Ph.D. Thesis, Oregon State University, 2009.

Joseph R. Shoenfield. *Mathematical Logic*. Addison-Wesley, 1967.

A. Signorini, A.M. Segre, and P.M. Polgreen. The use of Twitter to track levels of disease activity and public concern in the us during the influenza a h1n1 pandemic. *PLoS One*, 6(5), 2011.

V. Silenzio, P.R. Duberstein, W. Tang, N. Lu, X. Tu, and C.M. Homan. Connecting the invisible dots: Reaching lesbian, gay, and bisexual adolescents and young adults at risk for suicide through online social networks. *Social Science & Medicine*, 69(3):469–474, 2009.

P. Singla and P. Domingos. Discriminative training of Markov logic networks. In *Proceedings of the National Conference on Artificial Intelligence*, volume 20, page 868. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

P. Singla and P. Domingos. Markov logic in infinite domains. In *UAI-07*, 2007.

D.A. Smith and J. Eisner. Dependency parsing by belief propagation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 145–156. Association for Computational Linguistics, 2008.

J. Snow. *On the mode of communication of cholera*. John Churchill, 1855.

C. Song, Z. Qu, N. Blumm, and A.L. Barabási. Limits of predictability in human mobility. *Science*, 327(5968):1018, 2010.

S.A. Stouffer. Intervening opportunities: a theory relating mobility and distance. *American sociological review*, 5(6):845–867, 1940.

C. Sutton and A. McCallum. *An introduction to conditional random fields for relational learning*. Introduction to statistical relational learning. MIT Press, 2006.

K.P. Tang, J. Lin, J.I. Hong, D.P. Siewiorek, and N. Sadeh. Rethinking location sharing: exploring the implications of social-driven vs. purpose-driven location sharing. In *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pages 85–94. ACM, 2010.

Y. Tao, C. Faloutsos, D. Papadias, and B. Liu. Prediction and indexing of moving objects with unknown motion patterns. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 611–622. ACM, 2004.

Ben Taskar, Ming-Fai Wong, Pieter Abbeel, and Daphne Koller. Link prediction in relational data. In *in Neural Information Processing Systems*, 2003.

M.E. Tipping and C.M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society. Series B, Statistical Methodology*, pages 611–622, 1999.

S. Tran and L. Davis. Visual event modeling and recognition using Markov logic networks. In *Proceedings of the 10th European Conference on Computer Vision*, 2008.

Benedict I. Truman et al. Cdc health disparities and inequalities report. *Morbidity and Mortality Weekly Report*, 2011.

A. Tumasjan, T.O. Sprenger, P.G. Sandner, and I.M. Welpe. Predicting elections with Twitter: What 140 characters reveal about political sentiment. In *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media*, pages 178–185, 2010.

Naonori Ueda and Ryohei Nakano. Deterministic annealing em algorithm. *Neural Networks*, 11(2):271 – 282, 1998.

T.D. Ullman, C. Baker, O. Macindoe, O. Evans, N. Goodman, and J. Tenenbaum. Help or hinder: Bayesian models of social goal inference. In *Advances in Neural Information Processing Systems (NIPS)*, volume 22, 2010.

D.L. Vail and M.M. Veloso. Feature selection for activity recognition in multi-robot domains. In *Proceedings of AAAI*, volume 2008, 2008.

D.L. Vail. *Conditional random fields for activity recognition.* Ph.D. Thesis, Carnegie Mellon University, 2008.

M. Vlachos, P. Yu, and V. Castelli. On periodicity detection and structural periodic similarity. In *Proceedings of the Fifth SIAM International Conference on Data Mining*, volume 119, page 449. Society for Industrial Mathematics, 2005.

Jue Wang and Pedro Domingos. Hybrid Markov logic networks. In *Proceedings of the 23rd national conference on Artificial intelligence - Volume 2*, pages 1106–1111. AAAI Press, 2008.

B. Wellner, A. McCallum, F. Peng, and M. Hay. An integrated, conditional model of information extraction and coreference with application to citation matching. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 593–601. AUAI Press, 2004.

A. Wilson, A. Fern, S. Ray, and P. Tadepalli. Learning and transferring roles in multi-agent mdps. In *Proceedings of AAAI*, 2008.

A. Wilson, A. Fern, and P. Tadepalli. Bayesian role discovery for multi-agent reinforcement learning. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pages 1587–1588. International Foundation for Autonomous Agents and Multiagent Systems, 2010.

S. Wrobel. First order theory refinement. In *Advances in inductive logic programming*, pages 14–33. IOS Press, Amsterdam, 1996.

T. Wu, C. Lian, and J.Y. Hsu. Joint recognition of multiple concurrent activities using factorial conditional random fields. In *Proc. 22nd Conf. on Artificial Intelligence (AAAI-2007)*, 2007.

J. Yang, W. Wang, and P.S. Yu. Mining asynchronous periodic patterns in time series data. *IEEE Transactions on Knowledge and Data Engineering*, pages 613–628, 2003.

K. Yoshikawa, S. Riedel, M. Asahara, and Y. Matsumoto. Jointly identifying temporal relations with Markov logic. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 405–413. Association for Computational Linguistics, 2009.